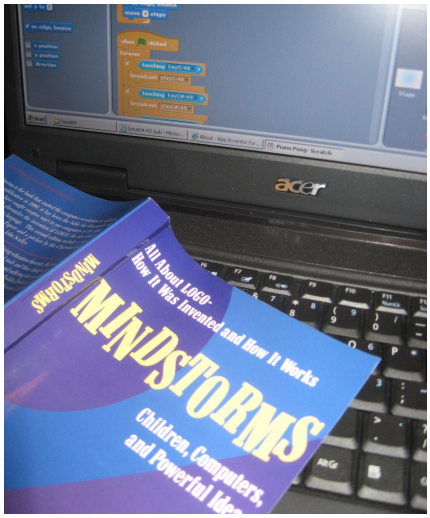
COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE
Part of BCS, The Chartered Institute for IT

SWITCHED ON

COMPUTING AT SCHOOL NEWSLETTER WINTER / SPRING 2015

CHILDREN, COMPUTING AND POWERFUL IDEAS



For several years, whilst lobbying for curriculum change, CAS had to focus on making the case for Computing as a distinct discipline. Of necessity, that meant patiently explaining what Computer Science meant and what it might look like as a school level subject. Computing is now here to stay and teachers need to consider not just what we will teach but also how and why it is taught. We are embarking on a pedagogical journey, exploring how best to teach the computational concepts that underpin the new subject.

Although Computer Science may be a new school discipline there is a rich school of thought on which we can draw. One of the benefits of thinking like a computer scientist lies in developing pupil's broader capacity to learn how to learn. That learning potential has inspired some of the game changing resources, such as the block based programming language, Scratch, that are freely available for us to use today. These, in turn, owe their origins to the pioneering work of Seymour Papert, best known as the father of Logo. Author of the seminal book Mindstorms, more than any other educational theorist, Papert makes a compelling case for the why and how of teaching Computer Science.

This issue has a special focus on his work. Getting to grips with the full potential of these 'tools for thought' is the exciting challenge facing todays ICT teachers. We will learn most by what we do, but in doing it, we can gain confidence from the fact that we are following in the footsteps of a remarkable visionary. We owe it to ourselves to become acquainted with his ideas.

INSIDE THIS ISSUE

The "Computing At School" group (CAS) is a membership association in partnership with BCS, The Chartered Institute for IT and supported by Microsoft, Google and others. It aims to support and promote the teaching of computing in UK schools.

bcs Academy of Computing

ISSN: 2050 -1277 (online) 2050 -1269 (print)

COLLECTIVELY CAS MEMBERS HAVE EATEN THE ELEPHANT!

Last August Simon Humphreys made a plea for help in categorising resources on the CAS Community. And so began an exercise in "crowdsourcing" …. Volunteers responding to the call got a few weeks in the limelight. The CAS Community home page now shows Recent Categorisers; the September list is shown. Not all CAS members will have the time and/or the expertise to create new resources so it's great that this sort of modest contribution to the community can be acknowledged. The categorisation counts expire after 3 weeks and so, by way of a permanently acknowledging good CAS citizenship, this discussion lists the proud owners of Gold (50+) and Platinum (300+) stars like this one. Our Ace Categorisers are: Katrina Morris, Dharini Krishnamoorthy, Andrea Keightley, Tim Eaglestone, Brian Lockwood, Peter Donaldson, Mark Tranter and Theresa Russell. Early on the 2nd November (hmm …. a Sunday) Andrea and Tim slugged it out for the last nibble of elephant; and Andrea won!

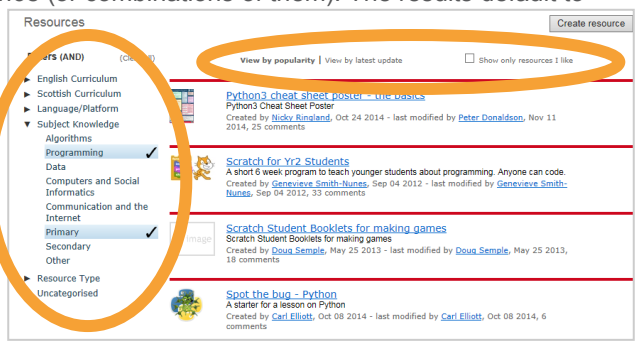
Recent Categorisers:
Dharini Krishnamoorthy (250)
Margaret Low (203)
Theresa Russell (142)
Katrina Morris (90)
Andrea Keightley (78)
Tim Eaglestone (73)
Paul Browning (72)
Brian Lockwood (71)
Peter Donaldson (58)
Miles Berry (38)

So that's the job done then?
Afraid not. Whilst all resources are now categorised, they may not consistently categorised. An informal working group, born out of the original crowdsourcing exercise, is forming to make recommendations to CAS as to the actions that would have the most impact for teachers.

MAKING THE MOST OF THE CAS COMMUNITY RESOURCES

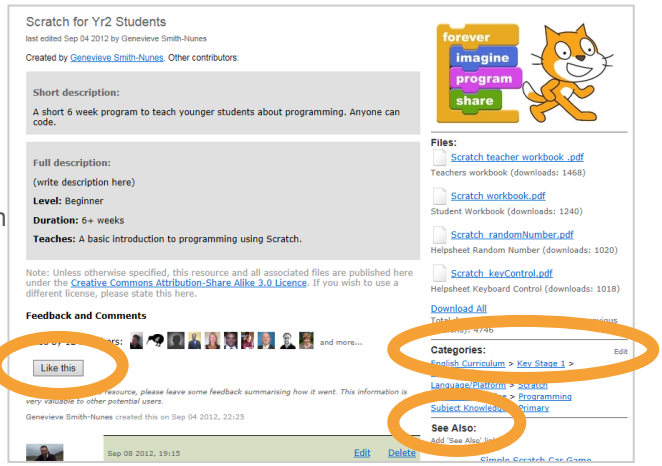
In the National CAS Survey last February, resources were valued the most useful aspect of the CAS Community, with discussion about approaches to teaching ranked second. So how can we enhance their impact for teachers?

The functionality of the CAS Community has seen some developments. As well as finding resources via the site search you can also filter by *resource categories* (or combinations of them). The results default to "View by popularity" making it easy to determine which resources other teachers find the most useful. How do you create a new resource?



Use the "Create resource" button on the main Resources page. The "Show only resources I like" check box also provides a useful personal bookmarking function.

When you drill down into a particular resource record you are in fact visiting a wiki page. Anyone can comment, edit or upload additional files to a resource page.



Moreover, you can very easily add value to a resource in just one click – by using the "Like this" button – or by adding a Comment. Other useful functions that anyone can use are "Edit Categories" (see sidebar) and add "See Also" links to signpost similar resources.

CAS MEMBERS: WHAT RESOURCES DO YOU NEED AND HOW CAN YOU HELP?

Clearly there are many resources but are all the curriculum bases covered with a reasonable balance? For example, confining ourselves to "data", we counted 100 resources with "binary" in the title, 13 with "hexadecimal" in the title but only 1 related to "data errors" when all these aspects are covered by the Subject knowledge requirements for entry into computer science teacher training.

Or are "meta-resources" (a resource bringing together a collection of other valued resources on the same topic) more useful as they would improve the signal-to-noise ratio for time-poor teachers? The "See Also" function could help by linking similar resources. Please use the CAS Discussion Forums to share your thoughts.

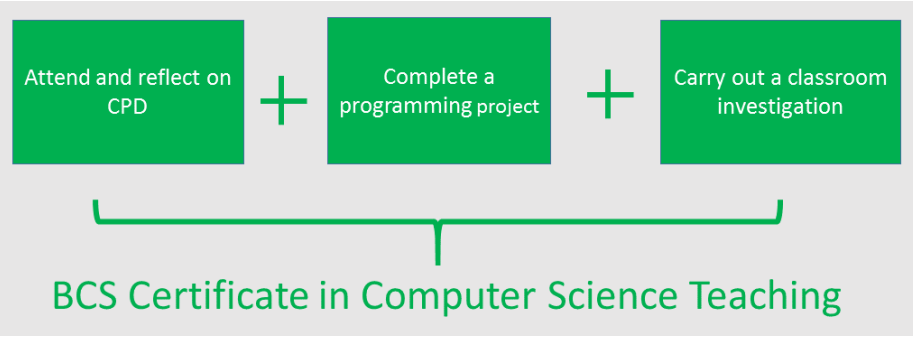
There are lots of ways to help too, with various levels of commitment, for example;

- Use the "Like this" or "Add comment" to let others know you've found a resource to be useful
- Contribute to improving and extending the existing categorisation effort
- Use "See Also" to link similar resources
- Create a resource but don't forget to add some categories too!

Paul Browning & Margaret Low

TEACHING COMPUTING? DO YOU GET THE CREDIT YOU DESERVE?

The BCS Certificate in Computer Science Teaching launched on 1st October and is now in full swing! The Certificate is intended to give you recognition of your development as a Computing teacher. Co-ordinator Sue Sentence explains how it works.



Whilst working towards the Certificate you will have the assistance of an e-assessor who will give you suggestions on how to improve on the work you submit. During the time you are enrolled on the Certificate, you design and develop a program that demonstrates your programming skills and also has some use to you as a teacher. This is because we believe that actually creating a working program gives you confidence in your own technical skills as well as demonstrating to your students that this is a useful skill to have, with some practical purpose. Quite often as a teacher we live in a world of little snippets of code demonstrating program structures and the project moves you beyond this. You will also carry out a pedagogical investigation on a chosen approach to teaching Computing, again with guidance from your e-assessor. This is to help you to develop your skills in teaching Computing, by exploring and reflecting on different strategies and focusing on how children learn the elements of Computing that are new in the curriculum.

There are versions available for both primary and secondary teachers. You are supported by an online learning environment where all the resource you need are housed and there is the opportunity to ask questions and talk to other Certificate teachers. It is a roll-on, roll-off programme, meaning that you can register at any time you like. We hope all Certificate teachers are members of CAS and engaged in the CAS community, attending Hub meetings and taking advantage of the low-cost Master Teacher training on offer.

"I have worked really hard over the past two years to increase my subject knowledge and become a better Computing teacher. This will give me credit for that and a very precise indicator of my skills as a computer science teacher."
Brendan Kenny, St Martin's School

We use a model of formative assessment, meaning that your e-assessor will support you throughout, looking at drafts of all the evidence you wish to present. You will need to incorporate their suggestions for improving work.

Most of you have been attending CPD to prepare yourselves for the new curriculum. The Certificate enables you to work towards projects that demonstrate what you have learned through the CPD and your experience of starting to teach Computing in the classroom. Our e-assessors are experienced academics who can give you tips and advice to develop your skills and knowledge as you progress towards the Certificate. Get the recognition you deserve. Interested? For more information go to http://computingatschool.org.uk/certificate.

CASE STUDY - SUNBURY HUB

The CAS Sunbury Hub, run by CAS Master Teacher Beverly Clarke (right), has now entered its second year. Sessions are well attended and are aimed at both secondary and primary colleagues. To give an idea of the topics discussed, the most recent hub event provided attendees with a variety of information. A presentation by Kingston University BCS Student Paul Hayward on how the BCS student chapters can support learners and teachers was well received. Beverly Clarke shared a presentation on developing enrichment in the new curriculum, an input on the BCS\CAS Teachers Certificate together with regular CAS news. Fellow CAS Master Teacher Deirdre Duffy hosted a session on the concepts of Sequence, Selection and Iteration and showed how this can be implemented in a visual programming language.

A second workshop was delivered by Jane Harding - a Raspberry Pi Certified Educator. Attendees engaged in a range of activities; how to setup and configure the Pi, read the GPIO pins, code music using Sonic Pi and code in Minecraft. They also benefitted from a demonstration using PiFace. All in all a very successful hub meeting which gave teachers valuable subject knowledge and practical skills. Once established, a hub provides a great pace for teachers to swap and share ideas, and thereby gain confidence.

GIVING IT A GO

Although I qualified as an ICT teacher, my enjoyment has always been with Computer Science. Before I started my teaching career I was a Network Technician in a school and the more technical aspects of my job were the ones that I found the most challenging, and therefore the most fun.

I felt there was very little challenge within the old IT curriculum and was therefore ecstatic when they announced that they would be changing it to the more challenging, problem solving, creative and industry focused computing direction. However now 4 years on from my previous career I knew it wouldn't do any harm to sharpen my skills. I had always been a keen member of CAS in the North East and had attended various hub meetings and training courses to get ready for the new curriculum. The master teacher course came onto my radar and I felt it was too good to be true. I had changed schools during my application and my new school were very encouraging to have me carry on with this.

The course had a great range of individuals in terms of experience in teaching, computing and various age ranges. We settled on completing our ten days over 4 windows that mixed up term time and weekends and in such a short term covered a wide range of information regarding course content, delivery and planning.

Since completing the course I have successfully applied to be Head of ICT & Computing where I teach at Sacred Heart High School, Fenham. I have certainly found my new role preparing to be a Level 2 Master Teacher a challenge, but one I am looking forward to. Working with other Master Teachers, and listening to the needs of staff in a range of other schools, with a range of experience developing the new curriculum has been something I continue to enjoy. I strongly believe in contributing to hub meetings and hope the courses that I am fortunate to deliver as a Master Teaching will help push forward a worthwhile curriculum that will benefit students for years to come.

MASTERING BEING A CAS MASTER TEACHER

If you love teaching and want to support others maybe Master Teaching is for you? Jo Hodge and Lee Willis recount their experiences completing their Level 1 Master Teacher year.

If you'd have said in January that I would be a Level 2 Master Teacher with Computing at Schools (CAS), I'd have laughed! Yet, here I am, and I don't regret a thing!

For the past 12 years I have been a teacher at Our Lady of Lourdes in Southport and an ICT co-ordinator for 8 of those. Being a person who has a great enthusiasm for all that is Computing, but with no specific ICT qualifications, I thought that I would be the last person they would want to do it. However, if you haven't already guessed, being a Master Teacher is not about that; it's about being prepared to offer support without necessarily being the ultimate expert. 'Teacher training teachers' is the excellent ethos endorsed by CAS, one I wholeheartedly agree with.

Sounds great, I hear you say? How do I become one? Easy! I registered an interest, complete an online application and was accepted as a Level 1 Master Teacher. From here off you go! You receive a full training package at an accredited university (mine was Sheffield Hallam) and enjoy a raft of computing challenges. These might take you out of your comfort zone a little, but nevertheless, they are well worth it.

So what's next for me? Currently, I am planning my three days of CPD which I need to deliver this year, as part of my CAS commitment. Also, I have decided to start my own hub in Sefton, due to the fact that the nearest Primary Hub is in Liverpool. This will hopefully offer additional support to all teachers within the local area and give everyone a platform to share good practise and new ideas. Networking is the key to being a good Master Teacher, so I have already met with some secondary and primary teachers, as well as my regional co-coordinator Carl Simmons, in Maghull to discuss their specific needs. Three Scratch basics workshops are in the pipeline, so I am feeling very excited! In my own school, my Head teacher has been extremely supportive (which has been invaluable) and just like everyone around the country, we are introducing the new curriculum. Therefore, I have organised training for staff on a half-termly basis to enable them to feel confident in their delivery of each element of the Computing curriculum.

For those feeling terrified at the prospect of binary numbers or algorithms, I would just like to say, don't be. This is the time to experiment, explore, try new things and maybe think about becoming a Master Teacher yourself. It has been an amazing journey thus far. I have had the opportunity to go back to university and meet some wonderful people. But my journey is not over and I foresee an exciting road ahead.

Whilst the current Master Teacher funding ends with the latest cohort appointed, CAS hope its success will make a compelling case for the DfE to extend and expand the scheme. If you're interested in becoming a Master Teacher, please keep an eye on, and download the overview from http://community.computingatschool.org.uk/resources/802.

THE WHY AND THE HOW OF THE NEW CURRICULUM

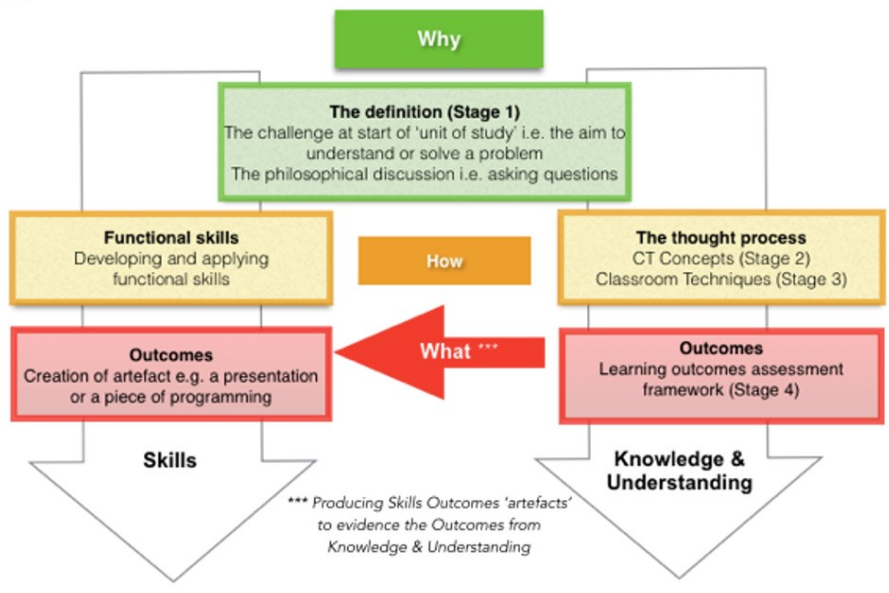
Much material supporting the old ICT curriculum focused on 'what' was being taught. CAS CPD Co-ordinator, Mark Dorling argues that the new curriculum requires a much greater focus on 'why' things are taught.

Thinking about 'how' and 'why' learners produced an artefact were often secondary considerations in the old ICT curriculum. The 'why' was often an assessment objective or exam qualification (in secondary schools) instead of a 'real-world' reason. A lack of focus on understanding the deeper 'how' and 'why' was highlighted in the influential Royal Society Report, Shutdown or Restart.

The over arching goal of the new national curriculum is to provide children across England with a high-quality computing education which could equip them "to use computational thinking and creativity to change the world". Changing the curriculum to focus on computational thinking and encouraging creativity is ambitious enough; changing the world is even more so! As educational observers have pointed out, a curriculum does not teach children, but teachers do. So how are teachers going to embed this goal in their classroom?

Computing's origins originally lie in philosophy and asking the 'why' question. I have often been reminded that the answer is only important if you ask the right questions! Computational thinking is a framework, not a recipe, for helping us to do this. Marketing methodology suggests that customers buy 'why' not 'what' and I believe this is no different in education. The over arching goal of engaging pupils is to empower them to take responsibility for their own learning through asking questions and understanding 'why' they are learning the topics.

Teachers can achieve this by refocusing their practice and considering the 'why' of the challenge they are setting for the learners at the outset. It is important to set appropriate challenges and select suitable hooks for learners. Pupils develop confidence by solving scalable problems. Making relevant links to other subjects and 'real life' can ensure engagement from girls, boys and minority groups.



Four stages of development in computational thinking are described in 'Developing computational thinking in the classroom' (see http://bit.ly/1jCUQl2). All four can be effectively mapped to the notion of focusing on the 'why', 'how' and 'what'.

THE CONTEXT IS KEY

In a primary school computing concepts don't have to be confined to computing lessons. They can be introduced in many contexts. Here's a couple of suggestions, by way of an example.

Your challenge is to program an animation to entertain an audience by recreating a dance routine from a music video using programming techniques such as sequence, iteration, procedures, selection and variables.

Your challenge is to act as a secret agent, who needs to send a classified message, via e-mail, to a fellow-spy in another country. Eavesdroppers can read messages on the internet. Therefore you need to encrypt messages to ensure that they can only be understood by your colleague. You will have to invent a cypher or secret code that only the two of you understand.

The 'how' of the subject is learners developing and applying a range of computational thinking concepts to understanding or solve the problem in hand. This is achieved through the teacher employing a range of computational thinking techniques. The 'what' is expressed in the evidence of the actual subject learning. The 'what' matches the learning outcome statements from the Computing Progression Pathways Framework.

We need to develop their IT skills at the same time. The challenge, of course, is the curriculum time. Where appropriate, learners should be using these skills (the how), to produce an artefact (the what). The artefact could be a video, podcast, presentation or piece of programming. The content should be based on the 'what' of the particular subject's knowledge, be it Geography, Science, Maths or even Computing.

The Barefoot Computing Project

Helping primary teachers get ready for the Computing curriculum

There are lots of new resources for primary teachers now online and many more still to come! At the time of writing we now have 16 teach-yourself concept resources and 17 exemplar activities to help you to learn and teach the computer science elements of the new computing curriculum.

A good starting point is seeing how our resources map to the computer science statements in the computing programme of study. For each statement we take you through the subject knowledge you need and give you a step-by-step exemplar activity to teach your pupils while deepening your own understanding about that topic. We have just released teach-yourself concepts for working with various forms of inputs and outputs. By the time you read this, materials for 'controlling physical systems' should have also been released. We have designed computational thinking key terms posters for your classroom walls. These and other resources can be downloaded from our website. Please do check the site regularly and follow us on twitter for the latest updates. You'll need to register, but it is all free.

Workshops for teachers

Our regional partners have been busy training volunteers to go into schools and deliver Barefoot workshops to primary teachers. Over 30 of these are currently advertised via the forthcoming workshops section of our website; however if you do not see one in your area, do contact your regional partner who can give you up-to-date information and arrange a workshop at your school if you wish.

Teacher forum

Do you have a question about teaching the computer science aspects of the new computing curriculum? If you do, you can use our teacher forum to ask the Barefoot team. We're keen to get you taking your first steps.

Barefoot Computing

STARTING FROM SCRATCH: MRS ELKES DIARY

In the first contribution to a regular diary, Lorna Elkes, Deputy Head at Brookmead School in Buckinghamshire shares her experiences starting from scratch.

I've been a teacher for enough years to be considered as 'experienced'. I am old enough to have experienced Computer Science at secondary school the first time round and have also worked within the IT industry - albeit many years ago. I basked in the pre-millennium shift towards using ICT in primary school. Like many I embraced interactive whiteboards and the vast array of new digital equipment as it arrived. I also trained others to use commercial, web-based technology and resources.

Then it all changed. The new primary curriculum offered great opportunities. A huge leap from endless PowerPoints; word documents combining text and images; or aimless web searching - and a chance to finally address programming. It has levered open the door to algorithms and firmly shoved in a wedge. Fantastic - at once I signed up for Scratch, installed Kodable on the school iPads and had school governors playing with Daisy the Dino. All was going swimmingly except then I got new job!

September 2014 and I am now working in a school with very little technology. Gasp! Horror! Some classrooms don't even have interactive whiteboards! There are a few working desktops computers but NO WIFI! Of course I knew all this when I applied then accepted my new post - and it has been fantastic. The lack of technology has created a clear vision. We are literally starting from scratch heading to Scratch!

Within weeks we had ordered a new learning platform enabling pupil and staff collaboration. Access too for parents and governors. Soon we'll be able to engage pupils at home via this new resource. Being cloud based there is no server to back-up, update or manage. Although we are not yet flush with computers we know this will change. We can plan for digital literacy using notebooks, iPads and of course laptops. We can develop the IT resources coherently, ensure the infrastructure is in place and shop around for suitable equipment. Last year the school's redevelopment of classrooms incorporated network readiness and work on the remaining parts of the school is due to begin.

Rather than shock and frustration, I find this very exciting with the new computing curriculum giving additional focus to our aims. I'm sure there are many schools whose current provision has developed piecemeal. We can plan for the technology we are already aware of but we can also include greater flexibility and the inevitable 'future proofing'. Of course we still need to address e-safety and the imminent planned changes are bringing more issues to the fore.

As for the curriculum requirements - having only limited access to computers has forced staff to look again. We have gone back to computing basics, thinking about what an algorithm really is. Thanks to the superb resources from Code-it such as their human crane, and other resources from Barefoot Computing, our school is beginning by laying firm foundations. I hope to provide an update on our journey in the next issue.

HERE BE DRAGONS! COMPUTING IN IMAGINATIVE CONTEXTS

There are many creative contexts through which children can encounter computing. Chris Leach, from Winchester House School in Brackley outlines a project he ran with Year 6.

Terrifying dragons, magical wizards and brave warriors may seem like they belong in an English classroom rather than in Computing but these all appear in adventures set in the Mythical Land of Ict which has helped bring concepts of computer science to our younger pupils.

The Here Be Dragons project was inspired by reading the excellent book, 'Computational Fairytales' by Jeremy Kubica (see right). After reading the book I felt it would be a fantastic way of introducing some computer science to my classes. As a result we have binary bonfire dragon warning systems and wizards that use algorithms rather than spells.

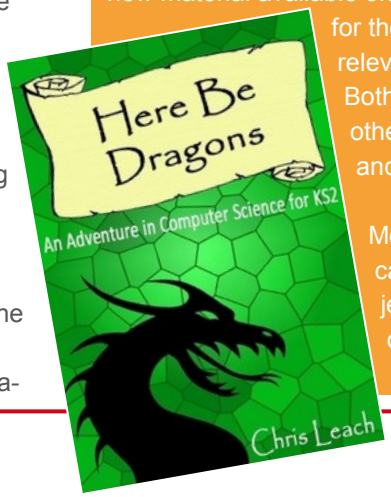
The adventure begins with a terrifying band of seven dragons attacking the remote villages of Ict. Each village is ordered to develop a system of signalling that dragons are nearby using bonfires to indicate when a dragon is close. Unfortunately due to health and safety laws each village is only allowed to have three bonfires lit so the villagers have to create a system for communicating that up to seven dragons have been spotted. This leads the children to creating a system based on binary numbers. The use of binary continues when each village is asked to keep an up to date census by using red and white flags to represent the village's up-to-date population.

As the number of dragons increase the King of Ict seeks to recruit a Royal Wizard to help defeat them. The abilities on the wizard's application forms are given a score in the style of Top Trumps and a Bubble Sort algorithm is used to select the best candidate for the job.

The Royal Wizard goes into battle with the dragons using randomly generated numbers and some pseudocode. Depending on the results of the wizard's algorithms the dragons were either tamed or left wild.

Eventually there were only fifteen dragons left. A binary search tree was created so each dragons can be easily identified. When they attack a village their individual weaknesses can be exploited.

The children loved this project and really enjoyed the mix of activities it featured. Not all of the project takes place at a computer as some of the tasks involved the children being sent off around the school looking for bonfires and dragon eggs. The first group of children to complete the project still talk about it nearly two years on and I am currently working on the sequel, Here Be Vampires, which will introduce more concepts of computer science and computational thinking.



COMPUTATION, SPELL DESIGN AND FAIRY TALES

"We all (I think) like to be immersed in tales of daring deeds. Sitting down to read a story is more interesting than reading a text book, but it's entertainment rather than learning. Computational Fairy Tales by Jeremy Kubica is one of those rare books that manages to cross the divide." So wrote teacher Lucy Bunce, reviewing the book for SWITCHEDON when it was first published in 2012. Kubica has a wonderful knack of introducing Computer Science concepts, even complex ones, in ways accessible to children of all ages. Primary teachers new to the ideas of Computing will love it, whilst A Level students will find much to illuminate their studies in an entertaining fashion.

A follow up volume, Best Practices of Spell Design introduces readers to good coding practices through the adventures of Marcus the wizard. It encourages readers to write well structured and documented code. Again, a marvellous book, packed with child friendly examples to take into the classroom. A tale to provide many lessons for students - lessons often learnt through their own painful experience.

Computational Fairy Tales are at http://bit.ly/1wHAlKj. This lists the stories by level, allowing a teacher to quickly select relevant chapters / posts. Best Practices of Spell Design is new material available only as a book. It has particular value for those just starting to code, probably relevant to secondary school teachers. Both are available from Amazon and others. See http://amzn.to/1zeRIp1 and http://amzn.to/1yGjXKq.

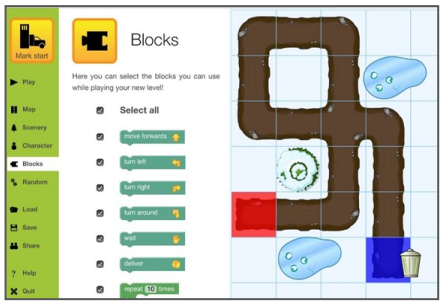
More details about Here Be Dragons can be found in the book of the project which includes photocopiable classroom material. Available through Lulu at http://bit.ly/1Gtc3rY.

INTRODUCING CODING USING OCADO'S RAPID ROUTER

CAS Master Teacher Ben Davies, a KS2 teacher at St Paul's CE Primary School in Withington, Manchester sings the praises of a free resource for primary schools.

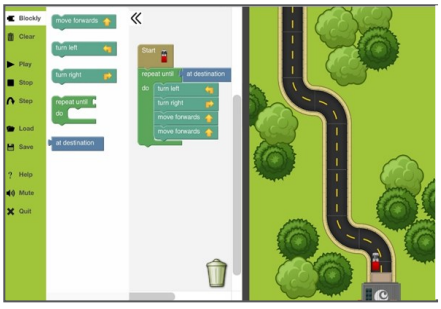
Rapid Router is the first program made available through Ocado's Code for Life project which aims to support teachers in inspiring the next generation of computer scientists. The software, which has been created by in-house developers, requires users to create programs using a visual programming language (Blockly) to guide the Ocado delivery van to its destination). Initially the product has a similar feel to other apps that give less confident teachers a starting point with Computer Science. However, delving further, reveals a product that is able to expand as the teacher and children's subject knowledge and confidence grows. Pupils are introduced to a key concept before writing programs that develop their understanding. As they progress, more challenging concepts such as repetition and selection are introduced. Embedded videos explain how these are relevant to Ocado's warehouses.

A comprehensive resource section includes printable versions of each level, aspect cards, vocabulary for display and two units of work focusing on algorithms and sequencing (key stage 1) and repetition and selection (Key Stage 2). Both units start with unplugged activities showing learning is based on understanding concepts rather than the software.



Where rapid router stands out from the crowd is the way users interact with the software. Individual pupil logins enable progress to be saved over a range of devices. Levels don't need to be completed sequentially, so teachers are able to select activities most appropriate to the lesson's learning intentions (instead of having pupils complete the precursory activities first). Rapid Router's crowning glory is the creator option whereby students (and teachers) can create their own routes. These can be shared with others. My class found this particularly engaging and were all attempting to create the most complex route. Children were soon creating routes to match given problems or include certain criteria (a conditional repeat) - real computational thinking.

Rapid Router is an excellent resource that will introduce pupils to the basic structures of programming. Furthermore it will support teachers' subject knowledge and allow them to guide children's learning rather than letting an app dictate it. With a Python expansion already in development and other products in the pipeline, Code for Life looks set to be an important tool in inspiring the next generation to become digital creators.



Safe. Together with O2

PRIMARY SOCIAL NETWORKING

Over the last year we have explored ways to engage and teach Computing. Leaving the programming / coding element to one side, a host of other areas contribute rich material for use in the new curriculum. One of the 'drier' areas was based around e-safety, ensuring children knew how to be safe online. Given the pace of digital learning is increasing, the tools we relied on were quickly outdated.

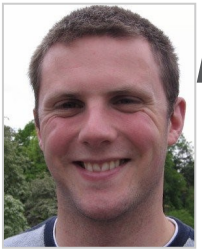
One way to teach this is through 'Safe Social Networking'. Primary age pupils are not allowed personal access to Facebook or Twitter but by the time they leave us we know that they are becoming aware of the possibilities of Social Networking. We feel it is part of our duty of care to prepare them to use these applications safely.

At Snowsfields Primary (like a number of other schools) we have been using the SoW from safesocialnetworking.org to guide children through the creation of profiles, blogging, sharing media and online collaboration. It not only meets the criteria set out in the Digital Literacy strand of the new curriculum but elements of the Information Technology strand too. The lessons can be applied to any Social Networking application – be it via a computer or using Mobile technology, so even if your young people are already using some form of networking tool, you can apply the SoW to suit your purpose.

For me, the beauty of using the Social Web, and ensuring children are taught how to use it appropriately, is that it can always run in the background of our learning at school – be this through the new Computing Curriculum, or in English, Maths or History for example.

Matt Rogers

TEACHING COMPUTING: ARE PUPILS ALWAYS BETTER CONNECTED?



Computational concepts can often be explained through activities away from computers. James Large, a CAS Master Teacher from Killigrew Primary in St Albans, urges you to go 'unplugged'.

"Are we going to do work on the computers today?" asked a Year 4 pupil. "Do you feel that we spend too much of our learning time at a computer? It sounds like you enjoy our unplugged lessons. Yes, there are a variety of ways we can learn about Computing." In fact, ever since taking over the planning and delivery of Computing at my school, I've witnessed the impact of unplugged lessons on children's learning. Barefoot Computing's (free) online resources are rich with 'disconnected' ideas including schemes of work, videos, plans and resources, as is Phil Bagge's website 'Junior Computer Science'. Almost every area of Primary Computer Science has an application away from a computer – and the children love it. More importantly for me, it's changed their ideas of what Computing is.

Interest rose when year 4 children came into their classroom to find the floor covered in red tape. It formed a series of 'courses' that the children had to walk, directed by their partner. This led on to learning basic Logo, then MSW Logo, and finally Scratch. Clear links to shape and space helped the children make the connection between Computing and Maths.

A ball of string was passed around the children to demonstrate and model the World Wide Web. In fact, on explaining that this is why it's called a web, there was the discernible sound of pennies dropping. Real-life links with their own experience help enormously. They role-played DNS servers making queries and linking with others. Some children suggested that this was like a shop or restaurant because you are asking for a very specific thing. Networks came alive.

Learning about databases in Year 5, we studied the property search website Right Move. There are many fields to search with and all kinds of housing to be purchased. Children were each given an individual budget. They took it in turns to role play the estate agent and the buyer, each time with a different brief, e.g. you are looking for no more than two beds, but you want a flat within 3 miles of Oxford station. It made them realise, in part, this is a very real job, done by very real people.

Are we always better connected? My answer: sometimes it's more fun to move away from the technology and understand where it's come from.

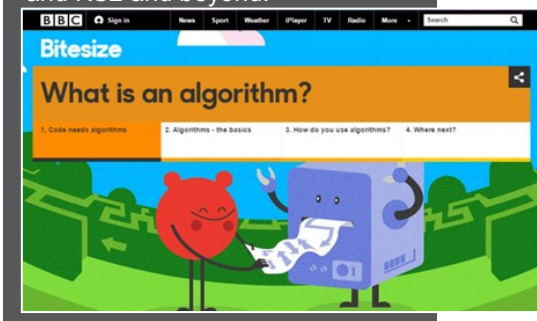
UNDERSTANDING THE NEW PROGRAMME OF STUDY


Simon Peyton-Jones

If you find some of the new National Curriculum for Computing a little daunting, then a new paper, written by CAS Chair Simon Peyton Jones will prove a valuable read. The document is a guide to the thinking behind the new Programme of Study for Computing in England. It was written for CAS members to help them unpack some of the rather dense language in the POS. Written in a style, using language accessible to teachers, it explains, with simple examples, the important concepts underpinning the new subject. Download from http://community.computingatschool.org.uk/resources/2936. This is a work in progress, so feel free to suggest improvements.

BBC BITESIZE COMPUTING

It has not been well publicised but the BBC has been updating its education resources and have quietly released new material for teaching Computing. All the resources have been broken down into the three strands of Computer Science, Digital Literacy and IT. There are sections for both KS1 and KS2 and beyond.



The guides are designed with children in mind. They are bright and colourful, arranged in clear sections which normally start with a question. Each has a low word count. Some have videos, interactive images and quizzes.

I used the KS2 videos 'How Does the Internet Work' and 'What is the World Wide Web'. They are particularly good at explaining complex ideas in simple terms.

The resources are ideal for use as an introduction for teaching new concepts or reviewing a topic. I think they will be valuable for teachers eager to brush up on some of their own knowledge too. The site will also be very useful as a tool to help extend the more able students in the class. I'd encourage everyone to take a look. Check out http://goo.gl/DtRcWD

Nic Hughes

UNIVERSITY TO HELP WITH SCHOOLS IT PROVISION

Schools are being invited to help shape the future of IT provision for Portsmouth pupils. The University of Portsmouth's IT department has won funding from the Higher Education Funding Council to find out what IT services need to be improved in schools and to see where the services and resources can be shared. The investigation is being run by the University of Portsmouth and Portsmouth City Council. The aim is to provide not-for-profit, sustainable and efficient IT services, whilst reducing costs and raising the quality of IT for pupils across the whole city. This is an opportunity to help shape the collaboration by providing feedback on the IT challenges faced by schools or by highlighting existing successes, innovative solutions and best practice that could help others.

It is being managed by Stuart Graves, from the university's IT Department. He said: 'Sharing services and the expertise available within the university is one way we can help others locally. We are already working with two schools as a pilot study to see what IT services they feel need improving. With the funding, we are able to improve them and make the services better. For example, one school might need better WiFi connections so we can look to do that.' The project is encouraging any interested parties to get in touch. The findings from this investigation and the proposed way forward will be shared with education providers at a showcase event at the University planned for early this year. Further details can be obtained from Stuart Graves. Email: stuart.graves@port.ac.uk

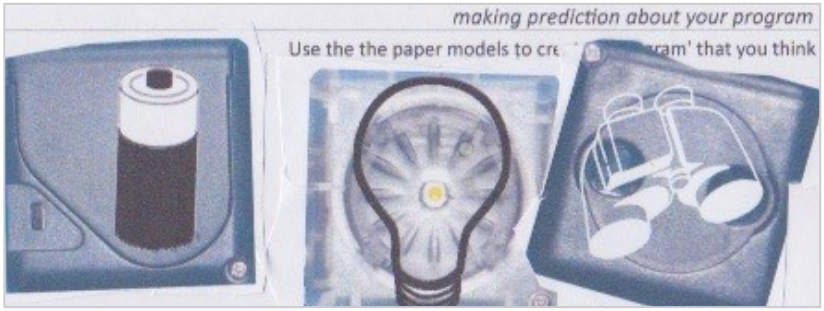
Tom Crick, CAS Wales, appeared on Blue Peter to publicise a new Dr Who game aimed at changing perceptions of computing and programming. You can play it at http://bbc.in/1tbk1E

EXPERIMENTS IN COMPUTER SCIENCE AT KEY STAGE ONE

As a new subject in primary schools, there is a real need for educational research into what works best with different age groups. Ben Wohl reports on a project he ran as part of his PhD research.

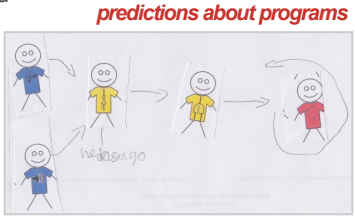
When it comes to the youngest pupils it is hard to know how to ensure the curriculum is on the right level but is also preparing pupils for the more complex computing they will encounter as they get older. Last June, as part of my research I conducted a study in three rural schools in north Cumbria. The aim was to explore different methods for KS1 pupils and teachers to engage with the new curriculum. Working with pupils and teachers brought home to me that the potential of the curriculum is not about technology at all. There are various ways of approaching the subject, both technical and non-technical but the power topic is in teaching 'computational thinking' to pupils at an early age.

Thinking like a 'computer scientist' is about teaching young people to view problems using terms like 'algorithm', 'logical prediction' and 'debugging'- the words used in the curriculum in fact. KS1 pupils are great at this. They get it! Algorithms are instructions, debugging is when the instructions go wrong, and logical prediction is guessing what is going to happen. Even better, these concepts can be taught across the curriculum, from reading to maths. Computing can be part of the whole school day. In my research, I focused on three methods of teaching Computer Science; unplugged activities, tangible computing using a system of blocks called 'cubelets' (used to make simple robots), and Scratch programming.



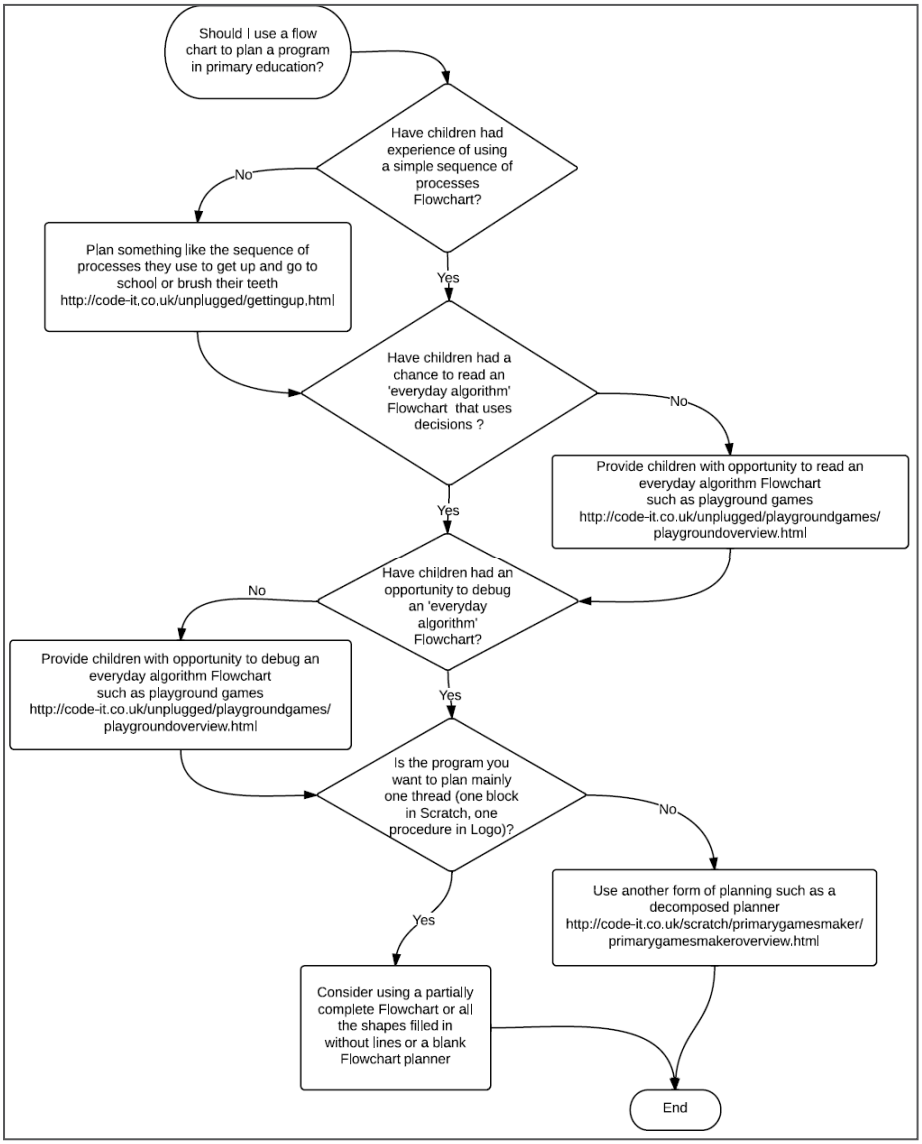
Using paper models to make predictions about programs

I was particularly impressed by unplugged activities for teaching underlying concepts. Cubelets were great for getting kids excited and involved but had their limitations. Scratch can be hard for this age group, but can allow them to explore simple programs. The most important thing to teach children in this new age is that computers can be programmed by anyone. Ordinary people can make computers solve problems and overcome challenges – you just need to know how to think like one. To support my project I posted a toolkit, Computer Science For Key Stage One to the CAS Community. A couple of teachers have commented, saying they feel some ideas are more suited to older pupils. One of the advantages of sharing ideas in the Community is the feedback it can generate. You can judge yourself by downloading the resources from http://bit.ly/1xSLO9V. The project has since won an award securing further funding. More details from b.wohl@lancaster.ac.uk.



PRIMARY PROGRAMMING AND FLOW CHARTS

Phil Bagge, Computing Advisor in Hampshire shares his thoughts on using flow charts for primary computing science.



This article first appeared on Phil's blog and is reprinted here with his permission. The original can be found at http://bit.ly/1B0njxA and the playground resources described are here: http://bit.ly/15lp3yT

My first encounter with flow charts was via Flowol, a programming tool that uses flow chart symbols to activate simple outputs such as lights and buzzers, controlled using inputs such as switches and sensors. Three years ago, I started experimenting with Year 4 pupils (8-9 years) creating simple flow diagrams to explain school procedures such as what happens when you are injured on the playground. Pupils were able to describe the processes verbally. We restricted ourselves to decision diamonds and process rectangles with start and stop terminators. If I am honest it was a real disaster. Pupils were able to sequence but few were able to create a working decision. The most common error was to create a sensible decision but to have both yes and no branches go to the same block. I wondered if the problem was that the process was unfamiliar so later tried a similar exercise with Year 3 and 4 pupils trying to chart getting up and going to school. Whilst more successful significant numbers still struggled with creating a working decision.

Children's reading skills are always higher than their writing. Perhaps pupils might need experience reading and working with fully formed flow charts before they could create one. I created five playground games flow charts. Pupils were not taught how to use them apart from being told to start at the start. The classes loved this activity and were able to work in small groups to decipher the flow charts. The only difficulty came with some pupils' reading level but mixed ability groups helped with this. Back inside the children were given an incorrect version of the flow chart. They worked in pairs or threes to try and debug the error. It was fascinating listening to conversations. In Year 4 all but one pair solved at least one problem and many two or three in the 25 minutes left. At the end I asked about strategies the group developed. Some said that they had each followed a different path once they reached a decision then reported back, others that they had all followed the same path so they could check each others' findings.

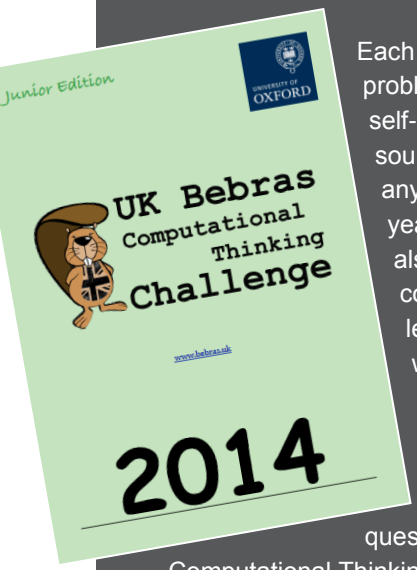
In some primary programming situations I can see the sense of using a flow diagram. A previously prepared flow chart could be shown to help pupils understand a more complex program. Pupils can complete a partially complete chart to help spot a programming pattern. They can work with simple programs which are linear. They are less use in programs that have multiple threads (lots of things at the same time) such as games programs with many characters.

Challenge pupils to create simple algorithms such as brushing your teeth using flow chart shapes. Progress to reading flow charts, followed by opportunities to debug them. Only use them to help plan a program after such opportunities and restrict to simple linear sequences. I always ask myself if I am adding an obstacle or a gateway before using a flowchart. Will it add complexity or aid understanding? My conclusion is summarized in the flowchart above.

UK Bebras Computational Thinking Challenge

After the successful pilot last year, the aim this year was to put this competition onto a firm footing so it can become a regular feature of the calendar for schools. To this end, sponsorship was generously provided by the Raspberry Pi foundation and ARM Holdings and Professor Peter Millican stepped up to be our man at Oxford University who has adopted this competition. And we have a new name that reflects the international nature of the competition.

This year we nearly doubled the number of students taking part (21,000 last year 39,000 this). We now have privacy, child protection and cookie policies in place and, using the feedback from last year, have improved the experience for schools in several other small ways. It was pleasing that we managed to produce a full set of problems for students that functioned well on all desktop browsers plus iPads. I was particularly pleased to be able to try a pilot of the younger age groups - down to six year olds who were also able to enter in small teams. This year's feedback has encouraged me to think that we can, with a few tweaks, make this a very worthwhile addition in the future.

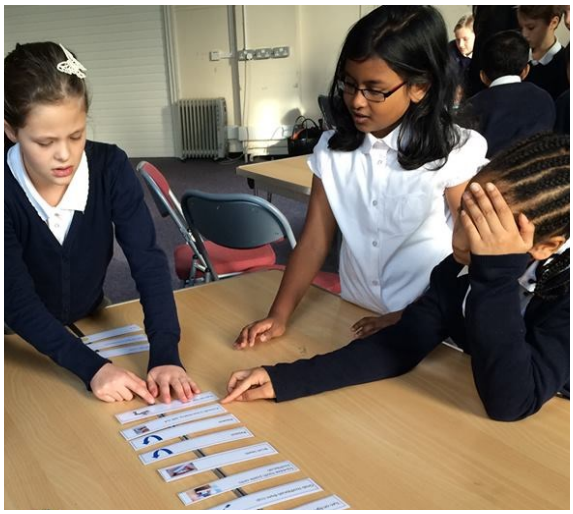


Each year's suite of problems becomes a self-marking resource available at any time during the year. We are now also providing competition booklets for schools which include answers, explanations plus a section showing how the questions relate to Computational Thinking skills. You can download the 2014 booklets from the front page of bebras.uk. The archived competitions can be found by clicking on the 'Preparation' tab. *Chris Roffey*

THE DIGITAL SCHOOLHOUSE: A MODEL FOR KS2/3 TRANSITION

An enrichment project bringing primary and secondary schools together has had an explosive start. Programme Director, Shahneila Saeed outlines the approach.

Since opening our doors in September, we have now worked with over 2100 pupils across London. Initial feedback from schools has been very positive, and primary schools are already booking *several* return visits for their pupils. So what is a Digital Schoolhouse? After the success of the original Digital Schoolhouse at Langley Grammar School, Ukie (UK Interactive Entertainment Association) and CAS joined forces to set up Digital Schoolhouse, London. With help from the Mayor's London Schools Excellence Fund 10 Digital Schoolhouses were launched across London linking educators with games companies and edu-tech social enterprises to create exciting lessons with real life scenarios to engage pupils.



Each Digital Schoolhouse is based in a secondary school and provides free enrichment days for visiting primary pupils. Visiting class teachers are provided with professional development opportunities through team-teaching and using lesson resources they help to adapt and create. The aim is for primary teachers to gain confidence to deliver similar lessons in their own schools. Digital Schoolhouse resources are freely shared and the model is designed to be flexible and responsive to pupil needs.

Already we're making a difference! Initial findings indicate that participation in the programme improves teaching pedagogy and pupil achievement. Lead teachers have reported that the training and workshop planning process have improved their understanding of pedagogy and had a positive impact on their teaching techniques across key stages. One teacher commented, "*There are lots of different ways of delivering a potentially dry topic in an interesting way...It's improved my teaching at KS4 and KS5*". The teacher also highlighted the inspiring techniques and resources being developed. With so much exciting work taking place, we've been busy gathering photos showcasing our pupils' work. All these and much more can be found on our new Facebook page, as well as regular updates via Twitter(@DigSchoolhouse). Visit www.digitalschoolhouse.org.uk to book a visit, download resources, or simply to find out more. If you are a secondary school and would like to set up a Digital Schoolhouse in your area then please do get in touch.

digital schoolhouse
Inspirational Computing for Kids

SEYMOUR PAPERT: COMPUTERS, KIDS AND POWERFUL IDEAS

Jack Lang, Chair of the Raspberry Pi Foundation, saw Seymour Papert's work first hand at MIT. He introduces our special feature looking at his ideas.

Seymour Papert's pioneering work with Logo (first developed in 1967) kickstarted an educational revolution. Papert's perspective, which has motivated many subsequent developments in educational computing, holds that programming can be a vehicle for engaging powerful ideas through active learning. Papert's thinking, which he called "constructionism", is a version of constructivist thinking based on the ideas of Jean Piaget, with whom he had studied. Constructivist thinking holds that individual learners construct or modify mental models to understand the world around them. Constructionism (with an N rather than a V) holds that learning can happen most effectively in a context where the learner is consciously engaged in constructing a public entity in the real world, whether that is a sand castle on the beach, or a theory of the universe in the lab. A teacher's role is thus that of a guide and a facilitator, rather than an instructor. Constructionist learning involves students drawing their own conclusions through creative experimentation and the making of social objects. Teaching "at" students is replaced by assisting them to understand, and help each other to understand by active exploration of a problem space.

It is the notion of active learning: kids learn best by experience—learning by doing, by exploring the world. Logo was intended to help teach maths, geometry and logic by creating a "mathland" that could be explored. Turtles came later, and it became obvious that they were a powerful tool to teach programming and new ways of thinking. Controlling a turtle drawing a line was a simple and easily understood analogy for sequential programming. Robotics, extending from controlling turtles, is a good way to teach computer science including real-time interactions, control theory, co-operating processes and modelling.

Logo, although easy to use, is a complete programming language. It was derived from LISP, the main language used for AI programming at the time, and that in turn was derived from lambda calculus. Logo has been described as "LISP without brackets". It is mostly text based, since displays then were uncommon and limited in function. In its turn Logo and Papert's work have been influential both in teaching and in computer science. The research group became part of the MIT Media Lab, who later developed Scratch. Papert's ideas were expanded in his book "Mindstorms: Children, Computers, and Powerful Ideas" first published in 1980.

Those were days when programming was thought of as highly skilled, available only to specialists. Yet Papert knew otherwise, as the children of faculty members and pupils from local schools, some as young as 9, found their way to the Media Lab and were indeed writing code. What Papert showed was that, given the right environment, almost anyone can program, and in doing so use it as a tool to explore new ideas and create new things.

> ***"The role of the teacher is to create the conditions for invention rather than provide ready-made knowledge."***

Seymour Papert

> ***"You can't teach people everything they need to know… The best you can do is to position them where they can find what they need to know when they need to know it."***

Seymour Papert's ideas were ahead of his time. He first talked about children using computers as instruments for thinking, learning and developing creativity in the sixties. The idea of an inexpensive personal device was then science fiction. But Papert was conducting serious research in his capacity as a professor at MIT. Over the years, this research led to many firsts. As he once noted, *"Every maker of video games knows something that the makers of curriculum don't seem to understand. You'll never see a video game being advertised as being easy. Kids who do not like school will tell you it's not because it's too hard. It's because it's boring."* As a new generation of teachers grapple with the potential embodied in his 'tools for thought' SWITCHEDON takes a look at his ideas and their importance in shaping a new pedagogy for the new Computing curriculum.

SEYMOUR PAPERT'S MINDSTORMS: THE POWERFUL IDEAS THAT LIE BEHIND LOGO

As our attention moves from 'what needs teaching' to 'how and why we teach it' Miles Berry shines the spotlight on the history of the ideas. That history is bound up with the remarkable achievements of one man, Seymour Papert; the father of the programming language Logo, and much, much more besides.

THE ORIGINS OF BIG IDEAS

Born in South Africa in 1928, Papert's childhood included plenty of 'tinkering' with cogs and gears, providing some early concrete experience of mathematical concepts. Papert studied maths at university before spending time working with Jean Piaget in Geneva. He worked in Artificial Intelligence research at MIT's AI lab alongside such heroes as Marvin Minsky and John McCarthy (inventor of LISP). It was there, in 1967, that Papert and others created the first version of Logo, as well as the first physical floor turtle robots which it would be used to program.

In Mindstorms (1980), Papert writes eloquently about his vision: *"In many schools today, the phrase 'computer-aided instruction' means making the computer teach the child. One might say the computer is being used to program the child. In my vision, the child programs the computer and, in doing so, both acquires a sense of mastery over a piece of the most modern and powerful technology and establishes an intimate contact with some of the deepest ideas from science, from mathematics, and from the art of intellectual model building."*

This pretty much sets out the CAS agenda, moving pupils from being consumers of digital content, to taking charge by writing their own programs. Why? Not as an end in itself, but rather because by doing so they would come to understand the world better.

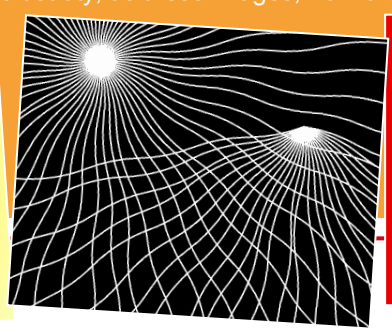
Computer Science as an entitlement for all, part of the national curriculum, is undoubtedly an achievement for which CAS and friends can be proud. We would, though, be wrong to think that this is new. Programming has been on the national curriculum since the very beginning in 1989, and children were learning to program in Logo well before that. Much of the very early work on teaching children to program was due to the influence of one man, Seymour Papert. Papert's insights into both the 'how' and the 'why' of school computer science education are as fresh and relevant as ever.

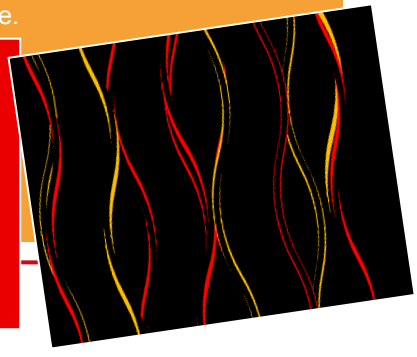
Papert writes about '**powerful ideas**', such as geometry and the laws of motion, which once grasped, change the way a child looks at the world around them. He argues that programming helps children grasp these ideas far more directly than other approaches. Programs are models, or 'microworlds' as Papert described them, which a child can create, adapt and explore. This is a big part of what national curriculum computing is setting out to do: 'to use computational thinking and creativity to understand and change the world'. Whilst Jeanette Wing deserves much credit for popularising 'computational thinking', the term was coined by Papert back in 1996, in relation to just this sort of powerful abstraction and exploratory 'tinkering'. Coding is a means to this end, not an end in itself.

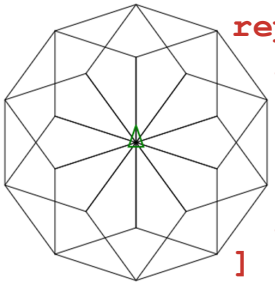
Logo is best known for its implementation of **turtle graphics**. The turtle was originally a simple robot, able to move forward and backward, or turn left or right through an angle, drawing as it moves. This is still a highly effective way to introduce pupils to programming (think Roamer Too or Pro-Bot), not least because it's so much easier to use logical reasoning to predict and debug instructions when you can *literally* 'play turtle', following the same instructions as the robot. Later the turtle became an on-screen representation, able to follow the same programs as the floor turtle. Logo also includes constructs for repetition, variables, selection and, crucially, procedures and functions. It would certainly suffice for meeting the national curriculum programming requirements for a text based language at Key Stage 3, although it's not as popular as a first teaching language as it once was. Visual languages remove the syntax barrier for young programmers, lowering the floor of entry. But their early development grew out of the pioneering work done by Papert. Those who designed them were standing on the shoulders of a giant.

THE BEAUTY OF COMPUTATION

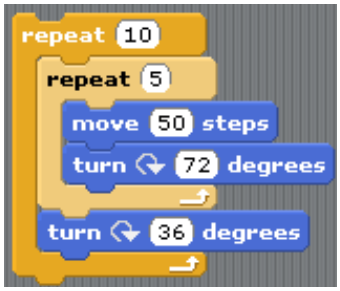
A central tenet of Papert's constructionism was to allow children to express their creativity through programming. There is something enduring about the aesthetic appreciation of geometric beauty, as these images, from the TurtleArt Gallery demonstrate.







```
repeat 10[
  repeat 5[
    fd 50
    rt 72
  ]
  rt 36
]
```



Logo's turtle graphics provides a great way in to the language. Children can experiment to find a way to draw familiar shapes or more complex geometric figures, such as the 'crystal flowers' of the old QCA schemes of work (see above). Through experimenting with turtle graphics programs, children can develop a better feel for geometry, ***discovering for themselves*** that 360° makes a full turn, the exterior angles of regular polygons, and so on. Papert writes about the importance of user-defined **procedures** in Logo, not merely as a convenient programming construct for decomposing complex problems, but as a means for a child to learn more about learning, as the child teaches the computer a new word, defining that using the words already in its language. Logo also supports functions and recursion, and indeed *could* be used as an introduction to functional programming.

```
to square :length
  repeat 4 [ fd :length rt 90 ]
end
```

```
to factorial :number
  if :number = 1 [output 1]
  output :number * factorial :number - 1
end
```

Given the 'powerful ideas' that turtle graphics provides access to, it's not surprising that it's been incorporated into so many of the programming languages used in schools. Scratch has a good implementation and the building block approach helps reduce the cognitive load of remembering commands and syntax. User-defined procedures only came to Scratch in version 2, which is surprising given Papert was Mitch Resnick's PhD supervisor. There are good turtle graphics libraries for Python, Small Basic and Touch Develop. Turtle System (overleaf) provides a carefully thought out structure to transition to textual languages. Moving from Scratch to any language is all the easier if pupils can see the connections between their Scratch scripts and their text based programs.

GOING BEYOND TURTLES

Lovely as turtle graphics are, there's so much more to Logo than just this. One of Papert's early papers (1971!) discusses using Logo to make a drill and practice maths quiz, in much the same way as we might get Year 4 to use Scratch to make this sort of 'educational game' now. He observed: "I have seen children for whom doing arithmetic would have been utterly boring, become passionately involved in writing programs to teach arithmetic, and in the pros and cons of criticisms of one another's programs."

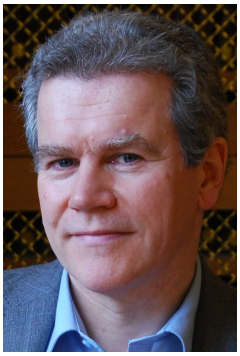
Brian Harvey's excellent Computer Science Logo Style: Symbolic Computing (1997) once used for undergraduate Computer Science at Berkeley, focusses on natural language processing in Logo. Part of the reason why programming back in the 80's failed to live up to its promise was that Logo became so closely associated with turtle graphics. Its potential as a general purpose programming language largely went unnoticed. Looking back in 2001, Richard Noss and others remarked: "Logo was marginalised, remoulded as harmless to the mainstream educational system, redrawn as 'a drawing tool' rather than as any kind of catalyst for rethinking the content of what is or what needs to be taught." It is up to teachers to ensure the renaissance of programming in schools doesn't suffer a similar fate this time round.

THE LEGACY OF PAPERT: THINKING ABOUT THINKING AND LEARNING TO LEARN

Just as there's far more to Logo than turtle graphics, so there's much more to Papert's work than Logo. His notion of 'constructionism' as a learning theory is at least as enduring as his work on Logo. **Constructionism** starts with Piaget's constructivism in that it sees each learner building and refining mental models of how the world works. Piaget saw learners as lone scientists, developing their understanding through discovery, exploration and experiment.

Papert sees learners rather as designers or engineers, recognising that the best way to develop mental models is through the *process* of building real models, shareable, public artefacts which embody something of that emergent understanding. Thus children write code not *merely* so we can assess their computational thinking but because it's through the process of writing code that computational thinking best develops. The sort of processes that lie at the heart of Papert's concept of 'learning through making' recognise the importance of pupils' own creativity.

Another theme that recurs throughout much of Papert's writing, and many of the YouTube clips of him talking, is 'passion', or even 'love'. Whilst these aren't comfortable words to use about schooling today, I think we can acknowledge that developing a love of knowledge, and a love of learning is something teachers and schools ought to do, even if this didn't make it onto the aims for the national curriculum. How can computing teachers best do that? Partly through allowing pupils to find something of the joy of creating code, but also through passing on something of their own passion; both for their subject and for learning itself.



USING THE TURTLE SYSTEM: AN EASY WAY TO START TEXT-BASED PROGRAMMING

***The Turtle System*, with teaching resources and coursework setting and marking tools, is available free thanks to a new project at Oxford University co-funded by the Department for Education. Peter Millican, Professor at Hertford College, Oxford, shows how to get started, both using the system and teaching with it.**

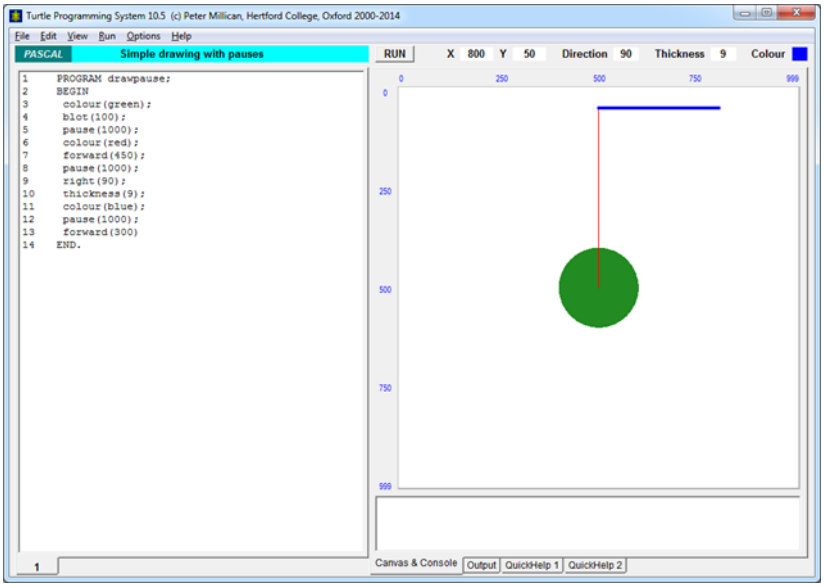
GETTING STARTED: HOW TO DOWNLOAD AND INSTALL

In the last issue of SWITCHEDON , I explained the principles behind *The Turtle System*, including its support of multiple "barebones" languages, simple but powerful graphics facilities, ease of setting up, precisely targeted error messages, and links to Computer Science through its use of a virtual *Turtle Machine* whose memory and machine code can be inspected in detail (and which enables *Turtle* applications to be run on the web and mobile devices as well as PCs). If any readers are interested in contributing to this project (either paid or unpaid), by helping to design follow-on course materials (which might be in specialist areas), please do let me know. I can be contacted at the address peter.millican@hertford.ox.ac.uk

To get the latest version, browse to www.turtle.ox.ac.uk and click on the "Download" button. If warned that it is potentially unsafe, do not worry: this is just because your computer doesn't recognise the program, but rest assured that nothing on this Oxford University site is malicious. Having downloaded the file ("TurtleSystem.exe"), you can put this on your desktop or within any folder on your computer or network, and run it from there. *The software will make no further changes to your computer system or network.*

Download the Turtle System
Windows Only

Having started the program, click on the "Help" menu; then within the "Examples 1" submenu, select "Simple drawing with pauses". This will load a short program in the system's Editor (at the left), and if you now click on the "RUN" button (top middle), you will see a picture being drawn on the Canvas (at the right) as shown below.



It's worth running this program several times to note exactly what it is doing and how the screen changes to reflect this. The invisible "turtle" starts in the middle of the 1000×1000 Canvas (where the X and Y coordinates are both 500), pointing North (i.e. direction 0º), and as it moves around in response to the commands "forward(450)", "right(90)" and "forward(300)", its position and direction are shown in the boxes at the top-right. Between movements, three "pause(1000)" commands tell it to pause for 1000 milliseconds each time, while three "colour" and one "thickness" commands tell it what kind of line to draw as it moves. The current Thickness and Colour settings are shown (in real time) in the boxes at the top-right, next to the "X", "Y", and "Direction" boxes. In this screenshot, we see the Pascal version of the program, which is currently best supported and particularly suitable for beginners, but there is also a Java version (available through "Languages" after selecting the "Power User Menu" option), with BASIC and Python also planned soon. Thus pupils can easily prepare for moving onto a variety of other systems.

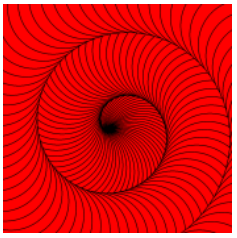
WebForum

TurtleOnline

This can be accessed at www.turtle.ox.ac.uk, where full documentation and a guided tour are available. Teachers can upload course materials, invite pupils to register for relevant courses, and have work submitted and collated online. Submitted *Turtle* programs can be viewed and run in a web browser, making assessment as convenient as possible. Moreover all this is available without having to prepare your own materials, since we have already commissioned ready-made courses from expert teachers, complete with PowerPoints, example programs, help resources, lesson plans, schemes of work, assessment criteria and targets, to provide all you need in the classroom and carefully tailored to "tick the boxes" on the new National Curriculum, as well as being engaging and educationally fulfilling.

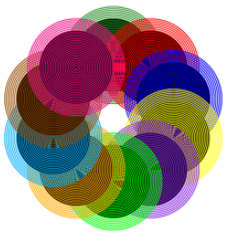
Turtle System is based on Turtle Graphics, an idea invented by Seymour Papert. This sort of programming, and the results it produces, are easy to understand because they are so immediately visual. But the Turtle System provided here shows that Papert's idea can go well beyond simple graphics, to provide a basis for fascinating and powerful programs that introduce fundamental concepts of software engineering and artificial intelligence.

```
PROGRAM forloop;
VAR count: integer;
BEGIN
 for count:=1 to 200 do
  begin
   forward(count/3);
   right(5);
   colour(red);
   blot(200);
   colour(black);
   circle(200)
  end
END.
```



Example 1

```
PROGRAM nestedloops;
VAR countblot: integer;
    countcirc: integer;
BEGIN
 penup;
 for countblot:=1 to 10 do
  begin
   forward(260);
   colour(black);
   blot(150);
   colour(countblot);
   for countcirc:=1 to 25 do
    circle(countcirc*8);
   back(260);
   right(36)
  end
END.
```

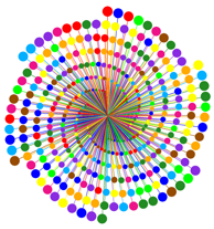


Example 2

```
PROGRAM parameterproc;
VAR count: integer;

  Procedure prong(len: integer);
  Begin
   forward(len);
   blot(len/20);
   back(len)
  End;

BEGIN
 for count:=360 downto 1 do
  begin
   randcol(10);
   prong(count+100);
   right(61)
  end
END.
```



Example 3

LEARNING FROM EXAMPLES

A very easy way for pupils (and teachers) to get used to the Turtle System and its possibilities is to play with some of the built-in example programs, running and editing them to see what happens. These start from straightforward "turtle graphics" drawing, then quickly bring in "for" loops to show how easy it is to create striking effects even with very short programs (see Example 1 above). Before long we meet nested loops (Example 2), then simple procedures (Example 3).

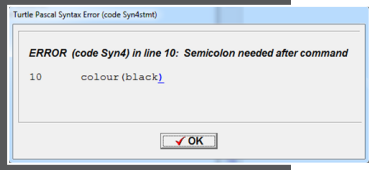
These are followed by examples illustrating all the main programming structures and commands, including "repeat" and "while" loops, recursive patterns, text printing and colour effects, keyboard and mouse input, mathematical and string functions, animation and simulation of physical systems, and various applications including video and strategy games, and cellular automaton models (e.g. the Game of Life).

Expression	Level	Count	Program Lines	Category
				Turtle: relative movement
back	1	1	8	
forward	1	1	6	
right	1	1	16	
	TOTAL:	3		
				Turtle: drawing shapes
blot	1	1	7	
	TOTAL:	1		
				Other Turtle commands
randcol	1	1	14	
	TOTAL:	1		
				Command structures
for	1	1	12	
procedure	2	1	4	
	TOTAL:	2		
				Subroutine calls
prong	0	1	15	
	TOTAL:	1		

The automatic program analysis for Example 3 above

SUPPORT FOR LEARNERS

As noted earlier, the *System*'s error messages are very precisely targeted, wherever possible giving a clear and specific instruction for remedying the error (as shown here) so that pupils can experiment and learn effectively without requiring constant supervision. Convenient illustrations and explanations of program structures or commands are provided through the two "QuickHelp" tabs beneath the Canvas, with "QuickHelp 1" giving 8 summary pages on the main features, while "QuickHelp 2" provides listings of the *Turtle* commands available. These are organised by level of difficulty (with just the simple commands shown initially) and by functional category (so users of any level can quickly identify needed functions). The "Edit" menu provides an "Auto-format" facility which neatly indents programs to a standard pattern, encouraging good practice and enabling "scope" errors to be identified easily.



SUPPORT FOR TEACHERS

The "Auto-format" also, of course, makes programs easier to mark. Moreover the system can easily be set up (through the "Power User Menu" option) to auto-format and/or run a program as soon as it is loaded – in which case examining a pupil's program requires no more than clicking on the relevant file. Another helpful feature is the "Usage" tab, which counts and lists the commands used in a program, organised by category (pictured left is the analysis of the "parameterproc" program—Example 3). I put these facilities into the original system when using it to introduce programming to large groups of "elective" students at Leeds University. By designing assessments accordingly (e.g. "write a program of at least 50 lines, using at least two looping structures …, to produce … effect"), it became possible to assess students' work extremely efficiently, freeing up a huge amount of time for face-to-face teaching rather than tedious marking.

GOOGLE LAUNCH CUSTOM COMPUTER SCIENCE SEARCH

Google announced the launch of Computer Science Custom Search, a customized search engine for finding computer science education resources. Developed using a collection of over 550 CS education websites, such as Khan Academy, Google CS First, Girl Develop It, Bootstrap, ScratchEd, Code.org and Made with Code to name a few, the CS Custom Search connects you to computer science education materials and programs. By focusing on a list of websites that provide primarily free and open CS education resources, the customized search engine ensures that you will find materials that can be readily adopted for your class, after school program, or enrichment for your child.

CS Custom Search has been designed to support a range of users with varying degrees of experience with CS. For those with extensive experience, CS Custom Search has been optimized to support queries for unique CS topics across a wide range of languages and platforms. For those with less experience, the custom search solution provides recommendations on a range of search topics to ensure that a lack of CS vocabulary doesn't negatively impact your results. Start by visiting www.cs4hs.com/resources/cscs.html. For more on Education programs at Google, visit www.google.com/edu.

Lissa Clayborn

COMPUTER SCIENCE FOR HIGH SCHOOL

The CS4HS program connects secondary school teachers with university academics, who can provide the training and tools needed to bring computer science and computational thinking into the classroom. Teachers gain the skills to develop their own computer science curriculum, helping to inspire a new generation of computer scientists. In 2014, four projects were funded in the UK and applications for 2015 funding will be accepted until February. To apply for a CS4HS Award, you must be affiliated with a university, technical college or an official non-profit organisation. For full eligibility criteria and more information, visit CS4HS.com.

ALTERNATIVE CONCEPTIONS AND WHAT CAN CAUSE THEM

You think they understand. Don't be too sure! Peter Donaldson, National Project Officer for PLAN C, identifies the roots of some common misunderstandings.

Teaching some of the concepts in programming can be a challenge but when they're busily working away after you've used a nice analogy or a perfect piece of live coding there's nothing quite like it. You think they've definitely got the idea and really understand variables. Then you ask them to explain what's happening in a simple bit of code with some assignment statements and the wheels fall off. Familiar feeling?

For example you might ask them what value gets printed out after the Python code shown right executes. The answer should be 4 which you think almost everyone will get but then reality hits. You get some giving the answer 9, others answer b and the final straw is when someone pipes up that they think it's 1! It would be easy to think they're just glancing at the code and randomly picking a number but if you ask them to explain their reasoning something much more interesting is usually going on.

```
Line 1  a = 9
Line 2  b = 4
Line 3  a = b
Line 4  b = 1
Line 5  print(a)
```

For the answer 9, the pupil explains that "a" is 9 and then "b" is 4. Then "b" is set to the value of "a" which means "b" is 9 and then "b" is 1. Aha, you think, when they see an assignment with variables on either side they're assuming it's carried out from left to right instead of right to left. When they choose "b" as the answer you find it's because they don't realise the value of a variable is fetched from memory unless it's on the left hand side of an assignment statement. Finally 1 is chosen because they think that Line 3 makes "a" and "b" point to the same memory location so when the value associated with "b" is changed on Line 4 "a" is also affected.

These are just some of the many alternative conceptions in programming that CS education researchers have uncovered over many decades. What unifies them is an attempt by novices to relate the code they see in front of them with the actions it causes the computer to carry out. During this process they need to use some form of background knowledge but because they don't have much experience of Computing they tend to try and draw from either everyday experience or other subjects they're more familiar with such as Maths and English. Unfortunately this approach can often lead them astray and leave them with a particular way of thinking that can prevent further progress in programming. Even analogies that we use to try and help them visualise these hidden mechanisms can sometimes backfire. For example the box model of variables can lead some novices to think that values are moved from one location to another during an assignment instead of being copied.

Most work on common misconceptions in programming has been done in higher education. We can learn a lot, but how it transfers to children will best be answered by teachers questioning and reflecting.

LEARNING TO PROGRAM WITH AN UNPLUGGED LANGUAGE

The best way to introduce youngsters to programming is away from a computer. Dave White, CAS Regional Coordinator for Essex and Herts argues for a focus on Computational Thinking.

This feature focuses on learning to program without a computer using UPL (an Unplugged Programming Language) Essentially this is similar to Logo or Python, so easily mapped. In UPL a pet/robot (in common with a sprite/ turtle) is defined by 2 characteristics:

- Its position - where it is standing if we are 'walking' the talk, or a point on the paper if we are drawing the path. We start at O (0, 0) the origin, which is usually centre stage/page (squared paper is helpful).
- The direction in which it is facing at any moment. We assume it is facing to the right at the start of a program.

There are 3 code instructions (in UPL) that the pet/robot obeys:

- **forward n:** our pet/robot goes forward n steps, drawing a trail as it goes. On paper a step is the length of the side of a square. We can go forward any number of steps e.g. **forward 3**.
- **left turn:** our pet/robot turns left through 90 degrees - just that, no movement forward
- **right turn:** similarly it turns right through 90 degrees

When we speak the instructions for walking the shape we use the full forms eg **forward2, left turn, right turn.** When we write the instructions, for convenience, we shorten them to **fd2, lt** and **rt.** We usually write the instructions in a program across the page, although we could write them one below another. A program (in UPL) is a set of code instructions taken from those above, which usually gets the pet/robot to move and draw.

A key pedagogical point to stress is that children need to be able to read before they write. The challenge is to crack the code, that is, figure out what our pet/robot draws when it obeys a UPL program. The challenges are shown right and the approach introduced below.

If your pet/robot obeys the shortened instructions, reading from left to right in each of the example programs below, what does it draw? Remember the pet/robot always starts at O facing right at the beginning of each program. It is worth looking for repeated patterns (symmetry). Writing repeated patterns vertically helps to pick out the symmetry. Look also for code that relates to physical components (decomposition) in order to simplify the programs. For example:

```
fd1 lt lt fd1 lt lt
```

can be written as:

A

```
fd1 lt lt
fd1 lt lt
```

(decomposition)

and represents the pet/robot drawing a straight line in the direction it is facing and returning to its starting point and direction. As we shall discover, this is a useful component program. We'll call a program that ends in this way a return program.

We can also introduce a control structure to take advantage of its symmetry: **repeat**

An instruction in the form

B

```
repeat 2 [fd1 lt lt]
```

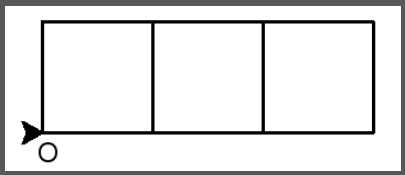
has the same effect as previously. In this instruction, the code inside the square brackets **[ ]** is obeyed twice. Can your pupils 'crack the code'?

CRACKING THE CODE

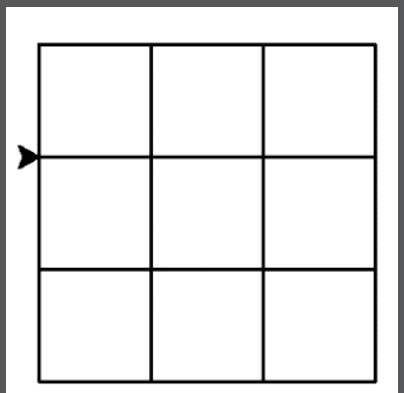
Crack the code in the following examples. Pick out the longest repeating pattern in the program and write the program(A) and program(B) versions for each to form a simpler program.

1) fd2 lt fd2 lt fd2 lt fd2 lt
2) fd1 lt lt fd1 rt fd1 lt lt fd1 rt fd1 lt lt fd1 rt fd1 lt lt fd1 rt
3) fd1 lt fd1 rt fd1 lt fd1 rt fd1 lt fd1 rt fd1 lt fd1 rt
4) fd2 lt fd2 lt fd2 lt fd2 lt fd2 fd2 lt fd2 lt fd2 lt fd2 lt fd2

How many are return programs?

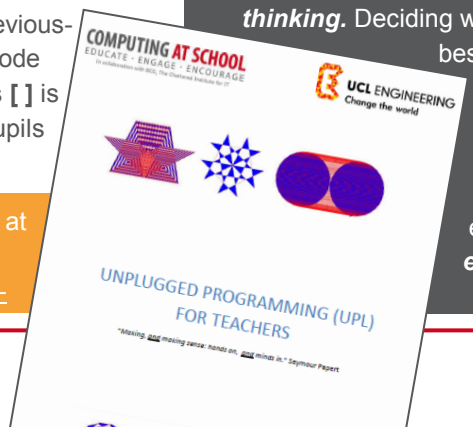


Write a program to draw the diagram above. Can you simplify it? Turn it into a return program?



Using what has gone before devise an algorithm to draw the pattern above. Here's one way; find out how you would draw the pattern freehand and get your pet/ robot to mimic you. Experiment with different ways. Each one is a different example of ***algorithmic thinking.*** Deciding which is best (short? clever? unusual? readable?) is an example of ***evaluation.***

These example are taken from material developed for a CAS course at UCL. For examples, discussion and booklet that develops the ideas much further please take a look at http://ispython.com/programming-



YRS
YOUNG REWIRED STATE
FESTIVAL OF CODE

Young Rewired State (YRS) is a global community. It is the junior, philanthropic arm of Rewired State. Since 2009, the YRS Festival of Code has been inspiring children. Running annually for a week in July, it encourages under-18s to team up and build their own projects. Each summer, regional centres open up in diverse venues, from companies, to schools and local authority buildings. Centre leads and mentors provide an environment where children can collaborate, experimenting, building, having fun and learning. It's a great opportunity to share ideas, discover new skills and create all kinds of projects with the proviso that each project must use open data. Many organisations offer support, including Ordnance Survey, who provided open source mapping data, the Met Office, Google and Twilio.

Children from eight to eighteen joined teams in their regional centres from Monday to Friday before relocating to Plymouth University for the competition. This year more than 180 teams representing over one thousand participants took part. Fifteen teams made presentations in the finals at Plymouth Pavilions in front of an audience of hundreds, and a panel of six industry judges. Children who had never presented to a group larger than their own class showed huge commitment and courage in stepping up and demonstrating what they had built. There was a jump in girls participating in 2014, accounting for around 30% of participants, and this is likely a trend which will continue. Ideas ranged from an intelligent Raspberry Pi-driven coat hook, to apps for reporting pot holes to councils, web sites to help students choose their best match of university, and CUDL, an app for crowd-sourcing help in an emergency. The weekend buzzes with ideas, debugging, indoor camping, pizza and ice-cream, and more debugging.

Finding suitable mentors and centre leads can be challenging; but if you have a week free at the end of July it is a great fun opportunity to support some extra-curricular development projects. If you have enthusiastic coding kids encourage them to join: it's inspirational, creative fun for them too.
http://www.yrs.io/ *Lyndsay Hope*

EXPLORE A WORLD OF BROWN DOGS AND BARBERS

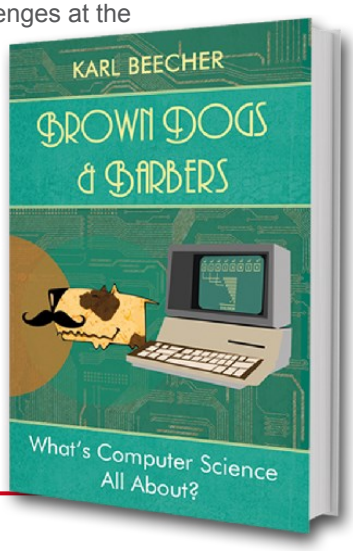
Have you ever studied Computer Science? If not, SWITCHEDON editor, Roger Davies, who teaches at Queen Elizabeth School, Cumbria urges you to read a new book.

What's Computer Science all about? Most people are able to give specific examples of things to do with Computing - networks, bits and binary, boolean logic, programming and so forth. But if, like me, you have no formal education in Computer Science it's difficult to begin to appreciate the discipline as a whole; how the different things fit together. Indeed, in the early years of CAS we had plenty of discussions about what the subject should look like at school level. Of course, now there is a National Curriculum and at higher levels, GCSE and A Level syllabi that specify what to teach. Unfortunately, subject specifications, aside from being as dry as a pot noodle, tend to break the discipline down, fragmenting rather than uniting it. If you want to grasp the deep concepts underlying the discipline, what's needed is a book that weaves a unifying thread through seemingly disparate themes.

Welcome to Karl Beecher's world of 'Brown Dogs and Barbers'. This book could not have come at a better time, nor be better pitched. Aimed squarely at the intelligent layperson, it requires no prior expertise and sits within the genre of popular science. Teachers, parents and older students will find it an invaluable introduction to key concepts and their practical application. It tells the story of how a new discipline, a new science was born. Arranged in six sections, each with chapters rarely more than five pages long, it is a book to engage even reluctant readers.

You can't understand where technology is going unless you appreciate where it came from. Karl Beecher recalls the often humorous history of computers in an easy, engaging style. He traces the pursuit of solutions to, at times, seemingly impossible problems. Most important, the technical advances are kept firmly in their place. The book is not really about technology at all - it's about the ideas and problems thrown up by its explosive development. As he eloquently shows, Computer Science was forged in the crucible of Maths rather than Physics. Chapter 1 will have you reasoning about the problem of computing a square root. By Chapter 2 you'll be pondering the semantics and logic of the Liar Paradox.

Brown Dogs and Barbers puts the cerebral challenges at the heart of the story of subsequent developments. There are bridges in Königsberg to cross, philosophers invited to dinner, traffic control problems to solve, jobs to schedule and message thieves to thwart. By the end of the book you not only have a real feel for the scope of the discipline but also a thorough grounding in the ideas that make your gadgets and networks tick. If you have no background in Computer Science, this book will be a revelation. And if you think you know what Computer Science is about, this book will invoke connections you'd never considered before. More details; browndogsandbarbers.com



GETTING TO GRIPS WITH THE IDEAS BEHIND BOOLEAN LOGIC

Some teachers may be unfamiliar with new terms introduced in the National Curriculum. Aidan Delaney, from The University of Brighton explains the simple ideas behind Boolean logic.

Software developers routinely encounter statements in Boolean logic. For example a school library may have three access levels; student, teacher and administrator. As a teacher and administrator, I can add and remove books. Non-administrator teachers can only search books, not add or remove them. When I log in, the application logic asks the database to "give me all users who are teachers **AND** administrators". The application checks the returned users and decides if I'm allowed to add a book, or not. Suppose we change the **AND** to an **OR** in the above query. Logically, the database returns a set of all teachers in addition to all administrators.

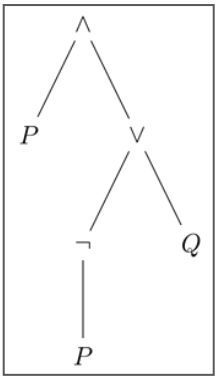
The difference between **AND** and **OR** (and we'll see **NOT** too) has wider impact than database queries. Such Boolean logic statements are found in the conditions of *if* statements and guards of *for* loops. To reason about program execution we need to evaluate formulas in Boolean logic. Such formulas evaluate to either **true** or **false** and are written in style that looks less like a programming language and more like maths.

As a nod to computational thinking we find that the syntax of Boolean logic is recursively defined. A formula in Boolean logic is defined as follows:

- each of a finite set of symbols is a formula in Boolean logic,
- if x is a formula in Boolean logic then (¬ x) is a formula in Boolean logic, and
- if x and y are formulas in Boolean logic then (x ∧ y) and (x ∨ y) are formulas in Boolean logic.

See the sidebar for an explanation of the symbols. This definition of the syntax of Boolean logic tells us that P, (P ∧ Q) and (P ∨ (¬ Q)) are formulas whilst ¬( )P and (P ∨ Q ∨ R) are not. Remember, every logic is a game and the only rules we have, so far, don't allow (P ∨ Q ∨ R) in the game.

Now we may ask if a formula evaluates to **true** or **false**. To evaluate the formula (P ∧ ((¬P) ∨ Q)) we start by evaluating the innermost parenthesis. This is more obvious if we look at the statement as a tree (right). The lowest P evaluates to **true** (by definition, propositional symbols evaluate to true). Working up the tree (¬P) is the inverse of P and thus evaluates to **false**. Moving to Q, which evaluates to **true** we now get to consider ((¬P) ∨ Q) which we can simplify to (false ∨ true) as we have previously evaluated the sub-expressions. By the truth table for ∨ (see sidebar) we see that ((¬P) ∨ Q) evaluates to **true**. Finally we evaluate the leftmost P to **true** (again, by definition) and we have to consider (true ∧ true) which evaluates to **true**. Therefore, the entire expression (P ∧ ((¬P) ∨ Q)) evaluates to **true**.



In programming we often replace propositional symbols with function calls. We might write file.is_open() & ! file.eof() to check if a file is open and not the end of the file. The function call file.is_open() will evaluate to either true or false. Notice that in the example code we have replaced the '∧' symbol with '&', the '¬' symbol with '!' and omitted the parenthesis. We have to map our knowledge of Boolean logic into the syntax of the programming language of our choice.

THE TRUTH TABLES FOR AND, OR AND NOT

The following truth tables define what "and", denoted using ∧, "or", denoted using ∨ and "not", denoted using ¬ actually mean. There are other symbols for "and", "or" and "not". Programmers in C-style languages use "&", "|" and "!" to signify "and", "or" and "not" respectively. Other text books use "•", "+" and " ¯ ". This is why it is important to define what is meant by the symbols (without a definition we don't have a logic!)

For ∧ we find that (p1 ∧ p2) is true only when both formulas p1 and p2 are true.

p1	p2	(p1∧ p2)
false	false	false
false	true	false
true	false	false
true	true	true

We find that the only case that (p1 ∨ p2) is false is if both formulas p1 and p2 evaluate to false. It's worth noting that the meaning of "or" in Boolean logic is not the same as the meaning of "or" in spoken English.

p1	p2	(p1 ∨ p2)
false	false	false
false	true	true
true	false	true
true	true	true

The meaning of "not" often catches out many students. However, (¬p1) simply evaluates to true whenever the formula p1 evaluates to false, and vice versa.

From the truth table for "not" you can see that (¬ (¬P)) evaluates to true as (¬P) evaluates to false and (¬(¬P)) evaluates to the opposite of (¬P) i.e. true.

p1	(¬ p1)
false	true
true	false

NORFOLK GIRLS COMPETE IN ALICE ANIMATION DAY

Following on from our successful Programming Challenge 4 Girls event last April, Jerome Thompson and I decided to put together a similar Transforming Education in Norfolk event. Teams from Attleborough, Wayland, City and Fakenham Academies each put forward up to six teams of girls. The event was hosted by City Academy and, on the day, organised by Fakenham staff. The girls were given a brief introduction to the Alice environment and then left

to explore and create on their own. After a short break they were set a programming challenge. With just one and half hours to complete it the time flew past. During the lunch break the girls were treated to boxes of pizza and lots of fizzy pop! While they tucked into their lunch, staff from the schools marked their animations.

After lunch, we had a short speech from Vicki Allen, founder of SyncDevelopHER a local group which aims to encourage the next generation of girls towards IT and computing. We were also joined by a Student Ambassador from the University of East Anglia. The Fakenham girls won six Gold and three Silver medals (– well done! Tamson Myhill, Lydia Fish and Molly Frost were also awarded a cup: *The Networking Innovation for Coding Excellence (NICE) Teamwork* (sponsored by UEA). The event was reported in the Eastern Daily Press (http://bitly.com/1ttHCKM) and some of the girls were interviewed on Radio Norfolk. A great day all round! *Sue Gray*

Positive role models can play an important part in shaping children's future expectations. Jen Fitzpatrick provides a peek into what it is like to be an engineer working at Google. The seven minute video, suitable for students, features interviews with several female engineers amongst others. Watch it at http://www.youtube.com/watch?v=wsnSBhWEyK4

CAS #INCLUDE DIVERSITY IN COMPUTING CONFERENCE

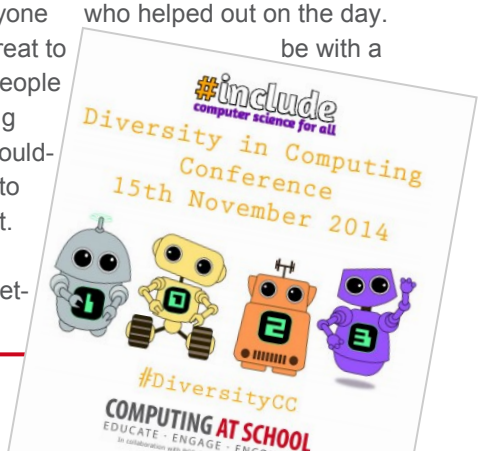
One Saturday in November, the school hall at Kingswinford High School was transformed as the CAS #Include team hosted their Diversity Conference. Emma-Ashley Liles reports on the discussions that followed.

After people had registered, grabbed a coffee and some awesome freebies from our sponsor, Google, Laura Dixon, #include chair, welcomed introduced Peter Kemp from Roehampton University who spoke about Social Mobility and Computing. He focused on the issues students from lower socio-economic backgrounds face in Computing education and raised the important question - is computing becoming an elitist industry? He also introduced us to Tommy Flowers - the man who designed and built Colossus but has achieved little lasting recognition due to his working class background. Inspired by this, many attendees vowed to introduce Tommy to their lessons.

Second on the bill was the fabulous Carrie-Anne Philbin, former teacher who now works as an education pioneer at Raspberry Pi. She asked 'Can Raspberry Pi be a tool for social mobility?' and spoke about the amazing things that happen when teachers, academics and industry experts work together. She played us some tunes in the form of a Sonic Pi program, composed (or maybe coded) by a competition entrant.

People headed off to chosen workshops, seven in total, each running twice. CAS #include covers five areas of diversity - gender, race, socio-economic status, SEN and disability. Each workshop focused on one particular area so attendees could specialize. The Dudley Computing teachers demonstrated as many engaging activities as they could in 60 mins. Catherine Elliott from the Sheffield City Council E-Learning Team showed how to adapt the computing curriculum for students with moderate learning difficulties. Matthew Parry from Swanwick School and Sports College showed us how to engage SEN students using Computing. Our own Rebecca Franks, who not only took the lead in organising the whole day, spoke about increasing the attainment of Pupil Premium students in Computing. Another #include committee member, Dawn Hewitson discussed reaching out to diverse communities. Lastly, I shared my own experience of becoming a software engineer exploring what educators can do to engage young women in the subject.

After Laura's closing remarks and a reminder that we are always looking for more volunteers, the competition winner for "Best Tweet" was announced. Congratulations Yasmin Allsop, the lucky winner of a copy of Carrie-Anne Philbin's 'Adventures with Raspberry Pi'. #include would like to say a huge thank you to everyone who helped out on the day. As one participant remarked, "It is great to be with a group of committed and dedicated people looking to make a difference to young people and their life chances". We couldn't agree more and we look forward to seeing more of you at our next event. Keep an eye on casinclude.org.uk/ where you can sign up for the newsletter and find out how to get involved.



A SPOTLIGHT ON PUPIL PREMIUM: SOME IDEAS FOR CLOSING THE GAP

At Durrington High School on the South Coast, work between the Computing department and Pupil Premium coordinators is helping close the achievement gap. Chloe Gardner discusses the range of strategies they have tried.

Whilst we want the best for all students, our focus on pupil premium students is closing the gap in achievement. We have tried a range of strategies. However, it is important to remember it is not a one size fits all approach. You must know your students and understand their strengths and weaknesses. You can address gaps in their knowledge but also praise their strengths. In 2013 the gap between PP and non-PP students at GCSE ICT was 53%. By 2014, this had dropped to 35%. Our Key stage 3 L6 gap reduced from 24% to 7% in the same period. We trialled a number of strategies, some developed ourselves and others shared by Rebecca Franks, Pupil Premium Officer at The Kingswinford School in Dudley.

The box below lists some of the strategies we use to engage our students. Our schemes of work interweave topics to ensure students are constantly recalling information and demonstrating their knowledge. It is still a work in progress though. Student learning is at the forefront of everything we do. We are proud to host the CAS West Sussex hub where colleagues at both Primary and Secondary levels gain confidence through sharing ideas. If you have suggestions for strategies used in your own school please contact me on cgardner15@durring.com

- **Raspberry Pi loan scheme** means students can rent out a Raspberry Pi kit to develop their skills at home and further improve their confidence
- **Extra-curricular programming clubs** focus on separate groups of students – those who have existing skills and those trying for the first time.
- **Local universities** helped set up visits for PP students to discuss career opportunities and the work that local IT and Engineering companies do.
- **Joining NACE,** the National Association for able Children in Education, helps you break down the pupil premium students. It allows you to focus on more able students as the PP student cohort are such a diverse group.
- **#include** events and blogs give lots of ideas and support.
- **Evening sessions** with PP students and their parents. Each year has their own evening. PP students projected to underachieve are invited to sessions designed to improve their work. Parents learn new skills too.
- **Personal Intervention Plans** sent home to parents of students who need focused support and reminders of the specific improvements they need to make. This could be theory or subject knowledge and topics or it could be skills in a particular piece of software.
- **Differentiated booklets** in lessons for students where needed helps support them with the use of key words, terms and activities to complete.
- **A Laptop loan scheme** for PP students is in the implementation stage.
- **Big Trak robots** engage students and get them thinking. This has worked for all abilities and seen more students engaging with programming.
- **In-school CPD sessions** raise awareness of strategies to ensure a whole school focus whilst evaluating their impact and cost-efficiency.
- **Computing Summer School.** Last summer we sent 9 PP students with low confidence on a FireTech summer school (http://firetechcamp.com/). They are now 'Computing champions' and buddy up in lessons with other students who are struggling and offer help and ideas to develop their work.

A CODING MISSION

New Edgehill epitomises the suburban outposts of the Reunited Kingdom. Set in the next century, the residents are interconnected via implanted chips to the neural net. Watched over by the benevolent Marshall Trent, who dragged the nation out of troubled times, social harmony is almost restored. Almost ….. When Cal Jones instinctively steps in to protect a young woman pursued by an Enforcer, his world turns upside down. The next 200 pages open his eyes to the true nature of society as he is drawn into the anarchists fight against Trent. This is a cracking adventure, in which Cal, Danni and Sasha battle the oppressive regime, that will have wide appeal.

What sets this book apart from others like it are a series of coding challenges interspersed with the story. The twelve challenges were written by Chris Roffey, well known for co-ordinating the Bebras Challenge. Each has background material, podcast instructions, strategy guides and hints. They require a beginners knowledge of Python. My guess is it will appeal most at Year 9/10 upwards (I bought it for my son), but judge for yourself. Each successful challenge unlocks the next. The contexts are excellent, integrating well with the storyline but also stand alone. A rich source of engaging problems accessed by free registration on the website; http://bit.ly/1mKCR3b I like this approach but have yet to try it in school. Encouraging youngsters in a club to read sections between tackling a weekly challenge might be one way, but I'm sure CAS members will share many more. *Roger Davies*

RASPBERRY JAM AND MAKER EVENT: TOP TIPS FOR SUCCESS

The task; to organise a day to find out about the world of coding, hacking and making. Having being inspired by a Mini Maker Faire at Manchester 's Museum of Science and Industry, I wanted to include a 'making' element. Fantastic events like this happen all over the country. So how was I going to make this one as good? Six months later, the National STEM Centre's Raspberry Jam and Maker Event drew 275 students, teachers and community members. Here's some ideas for running a similar event.

First look at the space you have available as this may limit the type of event you can organise. The National STEM Centre has an atrium (exhibition space), a lecture theatre and labs for workshops. I took to Twitter, Facebook, local CAS networks and Hack spaces to find experts and enthusiasts to support our day. Thank you to Alan O'Donohoe and Adrian McEwen for guidance and retweets! The exhibition space filled relatively quickly. School computing clubs are a great place to find willing volunteers. Ben Smith brought a small army of Y10/11 students, supporting the exhibition and leading workshops. We mailed D&T and Computing teachers within a two hour drive. We added the event to the Raspberry Jam and CAS events pages whilst local community groups also spread the word.

Get a good night's sleep beforehand! We asked volunteers to bring their own equipment and gave them time to set up before the doors opened. We made it clear where events were happening and the timings for the day. The more independent you can make the visitors, the easier management of the day will be.

My top tips are:

- leave time between workshops for moving
- post up signs so no-one gets lost! (toilets, workshops, exits...)
- provide Wi-Fi for visitors and exhibitors
- provide a Twitter feed for feedback
- have a team of people who know the grand plan and can help, the more the better!
- leave space to repeat popular workshops or book workshop places when selling tickets

This is by no means the definitive plan for organising events such as these, but I hope it gives you the confidence to try it out for yourself.

Gemma Taylor

TEACHERS JOIN FORCES WITH CODERS AT PYCON UK 2014

PyconUK (the annual community organised conference for Python users) has a flourishing education track. Nicholas Tollervey gives us a flavour of how the large cohort of teachers and pupils got on.

This year's PyconUK was the biggest ever with almost 500 attendees, of which 45 were teachers and 75 school aged pupils. It was the fourth year in a row we'd run an education track thanks to the support of our magnificent sponsors, Bank of America, the Python Software Foundation and the Raspberry Pi Foundation. Their generosity ensured we had cheap tickets for teachers and students and the cost of classroom cover, er, covered.

Our aim was to provide teachers with a day of accessible, friendly and practical collaboration with world class developers and education experts. To this end it was a pleasure to watch the education team from the Raspberry Pi Foundation, Minecraft maestro Martin O'Hanlon, Raspberry Jambassador Alan O'Donohoe, PyGame guru Dave Ames and robot overlord John Pinkney unleash their collective pedagogical forces on a room full of unsuspecting teachers, each of whom appeared to have adopted a tame Python developer for guidance (several of the developers worked for prestigious companies such as Google, Facebook, The Guardian and Bank of America). This was followed by practical workshops: teachers working alongside developers to solve problems, seek advice, create reusable resources and form networks of industry contacts. They also looked like they were having a lot of fun too: one group even produced an example solution for one of the trickier GCSE projects!

Martin O'Hanlon and Carrie-Anne Philbin, authors of the books below, at PyconUK

The following day teachers returned to try out their newly acquired skills on our 75 youngsters and observe various subject area experts lead workshops in a fun day of creative Pythonic coding. The end result being lots of tired but inspired young coders, each with a brand new Raspberry Pi to take home. It was also great to welcome so many teachers to the conference meal on Saturday evening and see so many at the "technical" conference on the Sunday. The UK's Python community is enthusiastic about education - especially when you consider the ovation Carrie-Anne Philbin received for her keynote presentation to the whole conference entitled "Miss Adventures in Raspberry Pi". We're already planning next year's event which should be bigger, better and more beneficial for all involved. We hope to see you there!

APPRENTICESHIPS: INSPIRING THE IT WORKFORCE OF TOMMORROW

New and exciting routes into the IT industry, through government funded apprenticeships are being left vacant. Many students are seemingly unaware of the opportunities available. Rob Chapman, from Firebrand Training explains.

500,000 missed opportunities by 2020! The UK tech industry is currently expanding at five times the rate of any other sector. With 1.5 million people already employed, a further 500,000 new people will be needed to fill IT jobs over the next 5 years. Despite the growing opportunities, here at Firebrand, one of a range of apprenticeship providers, we're experiencing a lack of applications for our IT Apprenticeship programs. If this trend continues, come 2020, many of these opportunities will be left unfilled. According to the EU there will exist a skills gap of 700,000 IT professionals across Europe by 2015 alone.

This problem does not limit itself to the Tech Industry. Technology has been intertwined with modern business. As a result, Computing is now vital across a huge range of sectors including farming, architecture and environmental science. This growing skills gap can affect multiple industries stifling economic growth.

Teachers are playing a part in changing that right now. There continues to be a misconception among children that IT is only for "geeks". What is it that makes youngsters want to be Hairdressers or Gym Instructors on low wages, rather than Ethical Hackers and Programmers earning upwards of £60,000? Teachers, play a key role in breaking down the "geek" stereotype. Their role is made even more important if we are to address the gender gap. Young women are still woefully under-represented with figures suggesting only 16% of the IT workforce are female. Teachers have the power to excite, inspire and above all educate the next generation of IT professionals. They can teach them to build mobile apps, make algorithms fun and educate about the varied, careers available within organisations like Facebook and Google.

They could also tell them about an IT apprenticeship. Computer Science at A Level or Degree Level need not be the only pathway for a young person. An apprenticeship also offers a gateway for those considering a career in IT. An IT apprenticeship is a real paid job. Apprentices are paid a minimum of £2.68 an hour; and typically earn a lot more than this. After qualifying, their starting salary can be upward of £20,000 with some earning more than £30,000 one year after completing their apprenticeship. For the period of up to a year, apprentices work in an IT focussed role gaining on the job experience. They receive training and achieve industry-recognised certifications alongside vocational qualifications. Such certification is invaluable. Created by the likes of Microsoft and Cisco, they demonstrate the skills required to work with a specific technology. While acronyms like CCENT or MTA may hold no meaning outside IT, in the industry they are badges of honour providing the stepping-stones to a lucrative career.

Moving forward, as teachers continue to develop the new curriculum, we hope they will begin breaking down that "geek" stereotype and in doing so inspire thousands with the idea that IT is for everyone. We hope they will tell their students that there are alternative career pathways and that they could avoid £30,000 of University debt. IT apprenticeships offer a real salary, industry recognised certifications and experience working in their future career. I'd encourage you to make your students aware of all the possibilities available to them.

THE FIRST RASPBERRY PICADEMY CYMRU

The first Raspbery Picademy Cymru was particularly apt since the Pi is now made in Wales. Indeed, the Picademy was hosted at Pencoed. Twenty six teachers gathered for two days of workshops, coding, building and even a tour of the factory floor to see the Raspberry Pi being made.

The team from Pi Towers ran workshops covering programming in Python and Scratch, building rainbow walkways by programming Minecraft, and using the GPIO pins in a range of implementations. There were workshops on breadboards, sensors, cameras and motors, while others made music using SonicPi. An array of quick and effective ideas that could be used in a class or club, together with lots of practical support for getting over technical bumps, made the event really productive. There was lots of opportunity to explore cross-curricular implementations, for example a bird box. When a bird flies in and out, sensors can activate a camera accordingly. A weather station with sensors feeding data back for analysis and integration, had potential for projects in Geography, Science or Maths.

The Raspberry Pi team aim to take a series of Picademies out around the UK in 2015, particularly to schools in the South West and the North, so keep an eye on the site for details of an event near you. Information and support: http://www.raspberrypi.org/.

Lyndsay Hope

DIGITAL WALES PROGRAMME OFFERS CPD FOR TEACHERS

A £370,000 contract from the Welsh Government, aimed at upskilling teachers in every secondary school throughout Wales in Computer Science, has been awarded to **Technocamps**. The computing outreach programme based at Swansea University has regional hubs at the Universities of Aberystwyth, Bangor, Glyndwr, the University of South Wales and Cardiff Metropolitan University.

Huw Lewis, the Welsh Assembly Minister for Education and Skills said: "Digital technology is only going to play an increasingly prominent role in all of our lives. If our future generations are going to be fully proficient using this new technology it's important that their teachers are equally comfortable with it."

With this contract, the Welsh Government's *Learning in Digital Wales* programme seeks to up-skill teachers by providing a taster session to a group of their pupils in a variety of different technologies, including LEGO Mindstorms, Scratch and Python. This will be through hands-on workshops delivered at their schools using teaching methods they will then be able to transfer to the classroom. Teachers will be encouraged to further enhance their skills through the **Technoteach** project which provides 22 hours of free CPD training at one of the University hubs. Over 100 teachers have enjoyed this free offering over the past year.

technocamps
Inspiring | Creative | Fun
Ysbrydoledig | Creadigol | Hwyl
Playground computing technoteach

Technocamps was founded in 2003, and since 2011 has delivered many inspirational computing workshops involving over 10,000 young people from 170 comprehensive schools across West Wales and the Valleys, as well as 50 primary schools throughout South West Wales with its **Playground Computing** project. Professor Faron Moller, Director of Technocamps at Swansea University, said, "I am delighted that this funding has come to fruition. It will allow Technocamps to broaden teachers' expertise across the whole of Wales. The new activities being planned will provide much needed encouragement and training in computer science to teachers."

LESSONS FROM RESEARCH IN COMPUTING EDUCATION

Sue Sentance reports on the recent Workshop in Primary and Secondary Computing Education (WIPSCE) conference in Berlin. The annual conference will be hosted in the UK next year!

WIPSCE attracts academics and teachers from all around the world, who share an interest in Computing Education research. It was held at the Free University of Berlin, the theme being "What do we know about our learners?" There were 64 participants from 15 countries, mostly from within Europe but including some from further afield including Mark Guzdial (Georgia Tech) and Susan Rodgers (Duke University) from USA who both gave keynotes. Tim Bell, well known for CS Unplugged, also attended from New Zealand. Many will know Paul Curzon of QMUL and instigator of CS4Fun. He gave a talk on the unplugged sessions running as part of Teaching London Computing. Quintin Cutts also spoke about Haggis, the new pseudocode being used in Scottish examinations, which generated a very lively discussion. Several PhD students presented on their research which is always very interesting and there were a range of posters on work in progress. We were fortunate to hear speakers from Poland, Bavaria, Cyprus and Israel where Computing has been established in the curriculum for many years. There are lots of lessons that we can learn from these countries. I was impressed by exciting initiatives happening in New York where only a small handful of schools across hundreds in the district offer any Computing at all.

I gave a paper about the CAS Network of Excellence and more specifically the work of the CAS Master Teachers. This contains evaluation data from the Master Teacher Training sessions - you can read this on the CAS Community website. There is a great deal of interest from other countries in the changes to the English curriculum.

On the middle day of the conference, local teachers were able to attend at a reduced rate. The exciting news is that next year the WIPSCE conference will be held in central London! This means that there will be an opportunity next November for teachers to attend and find out about the latest research in Computing education. I will post details on the CAS Community as soon as it is available.

CAS SCOOPS TOP EDUCATIONAL AWARD

In recognition of its initiatives promoting computing in primary and secondary schools, CAS won the 2014 Informatics Europe Best Practices in Education Award. The award is a prestigious recognition of a world-class initiative in computing education. Informatics Europe is the association of computer science departments and research laboratories. Simon Humphreys and Mark Dorling were presented with the award at the 10th European Computer Science Summit, in Wroclaw

BELFAST SCHOOLS GET CREATIVE FOR SUCCESSFUL EU CODE WEEK

Primary and Post-Primary schools across Belfast engaged in a series of basic coding, programming and games design workshops delivered by Nerve Belfast Creative Learning Centre. Irene Bell reports on an excellent week.

EU Code Week aims to expose people of all ages to coding allow them to explore it in a range of settings. The second annual event took place between11th and 17th October. Nerve Belfast is one of three Creative Learning Centres in Northern Ireland that receives government funding to develop the creative use of ICT and multimedia in education. The workshops allowed Nerve Belfast to promote the STEAM agenda across the key stages, by exploring a number of entry-level, age-appropriate technologies supporting the introduction of coding in the classroom.

Planning focused on the potential application of coding across the Northern Ireland curriculum, how it could be integrated into existing topics or subject areas, and how it could support effective teaching and learning. Jennifer Crotty, Education Manager at Nerve Belfast explains, "Workshops were carefully designed to introduce problem solving tasks within the context of coding, programming and games design. Every effort was made to ensure that tasks were manageable as classroom activities so they could be replicated and further explored by teaching staff in the future".

Primary school pupils designed their own computer games and animations using the Scratch Jr. iPad app, and programmed their own robots using Lego® WeDo® and Lego® Mindstorm Robots®. Heather Marshall, P7 teacher and ICT Coordinator from Cairnshill Primary School said, "It was great seeing the children being really hands on… not having everything work first time is really important for children to learn, you don't give up, you keep going… and seeing them have success at the end was super".

Post-Primary school students investigated conductivity using Scratch and Makey Makey™ kits and also developed their own computer games using Construct2 for PC and the GamePress iPad app. John McEvoy from Bloomfield Collegiate commented, "It's a wonderful opportunity. Hopefully we can build on this and use the activities we explored today to more effect".

The programme culminated in an interactive showcase event. One teacher remarked that it was "a great glimpse into the future of ICT and computing in the classroom". You can watch the EU Code week videos at www.nervebelfast.org.

TEACHING COMPUTER PROGRAMMING IN ITT

CAS member Ann O'Neill is coordinating and teaching the computing for all 600 ITE students in Stranmillis University College. Ann reports "The programme is now in its second year. All primary and post-primary student teachers acquire an understanding of the concepts of computer programming through Scratch and the Raspberry Pi. They learn how to create and teach a wide variety of Scratch projects, including Etch-a-Sketch, drawing regular polygons, interactive games and making music. With the Pi they write Python Turtle programs, complete a robot antenna project and traffic lights using Scratch GPIO, create a Scratch GPIO, make music with Sonic Pi and create Python programs in Minecraft. The students especially enjoyed simulating traffic lights with the proper colour sequence on the LEDs. This year, I have introduced a course in creating Android apps using App Inventor 2 to post-primary student teachers. I also offer each of these courses to teachers through the College's CPD programme". The next generation of Primary and Post-primary teachers are getting ready to spread their skills throughout the schools in Northern Ireland.

GOOGLE CREATIVE TECHNOLOGY PRIZE

Under the guidance of CAS teacher Tim Gamble the Dalriada Robotics club competed against other UK schools in the *National Science & Engineering Competition*. Tim says "Over two exhausting days, our Robotics team exhibited their project and made presentations to judges from various industries. One particular set of judges from Google took a special interest in the project and awarded them the prize for '*Creative Technology*' and £500! Needless to say, the team were ecstatic and, to top it off, Google gave them a special invitation to visit either of their headquarters in London or Dublin. Well deserved congratulations go to Oliver Warke, Finlay Roulston, Claire Mulholland and James Hancock from Dalriada School in Ballymoney.

A PAUSE FOR THOUGHT

The term 'Internet of Things' (IoT) was first coined in 1999 and describes the exponentially expanding network of internet-enabled devices. It has vast potential, allowing our devices at home, at work and in between to talk to each other and to third parties. These devices, usually with sensors collecting data to transmit to other devices, are burgeoning. Examples range from remotely controlling our central heating and health care gadgets able to transmit readings to medics, to large scale applications enabling shipment tracking. The IoT promises environmental benefits as smart meters on a smart grid enable monitoring and more responsive energy management. Some research predicts 30bn devices will be connected to the IoT by 2020.

So, it opens a vista of possibilities, but also poses unique challenges. Each connecting device is a new potential exploit. With increased connectivity, the opportunity for disruption increases. Headline hacks include pacemakers, cars, smart-light bulbs and baby monitors. Even smart-fridges have been used to send spam. How often do we connect new devices without considering the risk in a way we would never do with a desktop computer? With '123456' as 2013's most popular password, can we trust our children to secure the IoT? And even if we consider security, do we know what happens to the data being harvested?

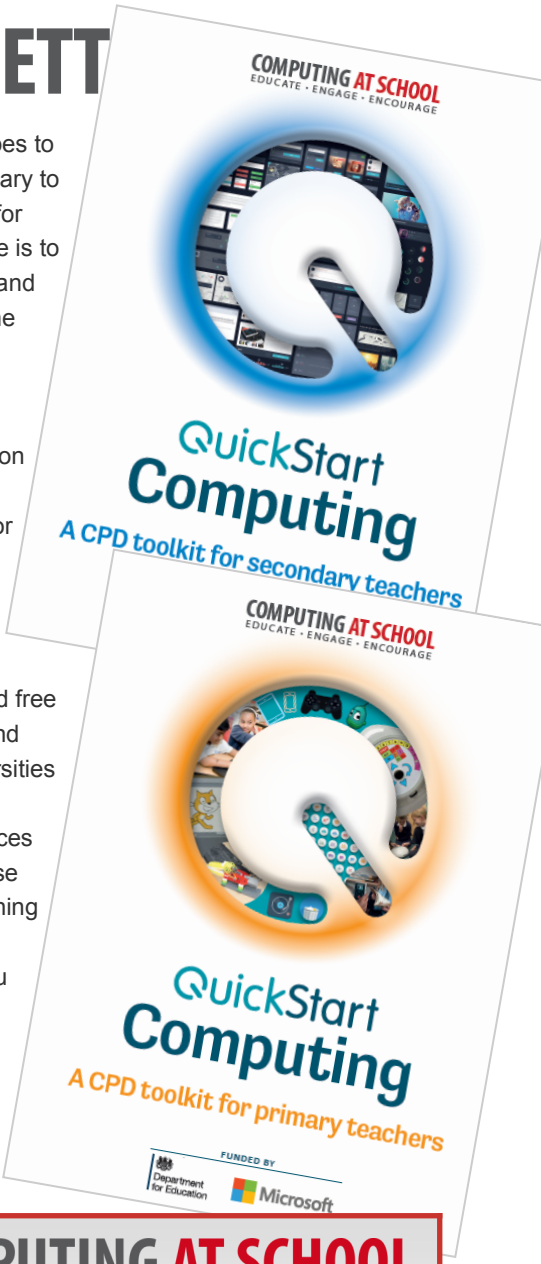
As standards and protocols develop, the security of devices and our data must be paramount; together with raising awareness amongst users of the potential benefits and pitfalls. For our students, there will be work for years to come in the challenging field of security in the IoT. *Lyndsay Hope*

A QUICK START TO COMPUTING LAUNCHES AT BETT

The CAS QuickStart Computing project hopes to provide teachers with the resources necessary to successfully run a Computing CPD course for their colleagues. The aim of the CPD course is to help empower teachers to design, develop and deliver a computing curriculum tailored to the specific needs of their school.

The CPD course is split into two parts, one being focused on primary and one focused on secondary. They are not mutually exclusive though. You may well find both are useful for them and both are free for teachers in both the primary and secondary sectors.

Hard copies of the QuickStart toolkit will be available at BETT but will also be distributed free through local CAS Hubs across England, and also through CAS Lead Schools and Universities in the CAS Network of Excellence. In total 40,000 free copies of the QuickStart resources will be distributed throughout England. These include printed materials and a DVD containing all the resources as well as videos covering the main CPD topics. Please note that if you have trouble getting a physical copy, digital copies of all the resources are also freely available to download for teachers anywhere in the UK. See the CAS website for more details.



COMPUTING AT SCHOOL
EDUCATE · ENGAGE · ENCOURAGE

Computing At School was born out of our excitement with the discipline, combined with a serious concern that students are being turned off computing by a combination of factors. **SWITCHEDON** is published each term. We welcome comments, suggestions and items for inclusion in future issues. Our goal is to put the fun back into computing at school. Will you help us? Send contributions to **newsletter@computingatschool.org.uk**

Many thanks to the following for help and information in this issue: *Phil Bagge, Irene Bell, Miles Berry, Paul Browning, Rob Chapman, Beverly Clarke, Lissa Clayborn, Claire Davenport, Ben Davies, Roger Davies, Aidan Delaney, Peter Donaldson, Mark Dorling, Lorna Elkes, Chloe Gardner, Stuart Graves, Sue Gray, Jo Hodge, Lyndsay Hope, Nic Hughes, Catriona Lambeth, Jack Lang, James Large, Chris Leach, Emma-Ashley Liles, Margaret Low, Peter Millican, Faron Moller, Chris Roffey, Matt Rogers, Shahneila Saeed, Sue Sentance, Andrew Shields, John Stout, Gemma Taylor, Nicholas Tollervey, Dave White, Lee Willis, Ben Wohl.*

www.computingatschool.org.uk

Computing At School are supported and endorsed by:

bcs The Chartered Institute for IT
Enabling the information society

Microsoft Research

Google

CPHC
The Council of Professors and Heads of Computing